TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

JANNE KLAPER

TIETOTURVAN HUOMIOIMINEN MAKSUJÄRJESTELMÄSSÄ

Diplomityö

Tarkastaja: professori Tommi Mikkonen
Tarkastaja ja aihe hyväksytty
Tieto- ja sähkötekniikan
tiedekuntaneuvoston kokouksessa 3.
maaliskuuta 2015

TIIVISTELMÄ

Janne Klaper: Tietoturvallisuuden huomioiminen maksujärjestelmässä
Tampereen teknillinen yliopisto
Diplomityö, 59 sivua
Toukokuu 2016
Tietotekniikan diplomi-insinöörin tutkinto-ohjelma
Pääaine: Ohjelmistotuotanto
Tarkastaja: professori Tommi Mikkonen

Avainsanat: tietoturva, maksujärjestelmä, varmennus, tietokanta, verkko, palvelinohjelmisto

Tietoturvallisuuden merkitys korostuu nykyisessä verkottuneessa maailmassa entistä enemmän. Samaan aikaan digitalisoituva maailma kaipaa paljon uusia ohjelmistoja. Nämä ohjelmistot joudutaan usein tekemään kiireellä, jolloin tärkeiden asioiden painottaminen oikein tulee merkityksellisemmäksi. Rahaohjelmistoissa on erityisen tärkeää huolehtia ohjelmiston turvallisuudesta rahan kriittisen luonteen takia.

Tämän työn tarkoituksena on tarjota ohje maksujärjestelmäprojektiin, jotta kiireisessä aikataulussa osattaisiin panostaa olennaisiin ja tärkeisiin asioihin tietoturvassa. Kun ohjelmistoprojektissa huomioidaan työssä käsitellyt asia, saadaan varmistettua hyvä pohja järjestelmän tietoturvallisuudelle. Monet työssä esiin tulleista asioista koskevat myös muita järjestelmiä. Tämä diplomityö keskittyy erityisesti maksujärjestelmiin, koska ne ovat kriittisiä järjestelmiä ja tarvitsevat korkean saatavuuden. Erityinen paino on maksujen varmentamisesta vastaavalla palvelulla, joka on hyvin kriittinen osa toimivaa järjestelmää.

Työ käsittelee järjestelmän kolmea eri osa-aluetta, jotka yhdessä määrittelevät järjestelmän tietoturvallisuuden. Ensimmäisenä osa-alueena on verkkoinfrastruktuuri, joka luo pohjan järjestelmän toiminnalle. Järjestelmä kommunikoi muiden järjestelmien kanssa verkon avulla, ja mahdollistaa järjestelmän hajauttamisen. Toisena osa-alueena on tietokanta, joka on järjestelmän tietojen keskipiste. Tietokannan toimivuus takaa tietojen ylläpitämisen ja oikeellisuuden. Kolmantena osa-alueena ovat palvelinohjelmistot, jotka muodostavat järjestelmän toiminnallisuuden. Näiden osa-alueiden sisällä tarkastellaan tietoturvan kolmea keskeistä osaa, jotka ovat luottamuksellisuus, eheys ja saatavuus.

Arvioinnissa käydään läpi sitä, miten toteutettu maksujärjestelmä onnistui tietoturvallisuuden kannalta. Tietoturvaan liittyy aina monia asioita, mutta muun muassa järjestelmän suorituskykyä on saatu parannettua merkittävästi käyttöönoton jälkeen. Tietoturva on ollut järjestelmässä hyvä, ja sitäkin on parannettu auditointien antamien suositusten mukaan. Kaiken tämän mahdollistavat hyvät ohjelmointitavat sekä kattava testaus, joiden ansiosta muutoksien tekeminen on helppoa.

ABSTRACT

Janne Klaper: On the Importance of Information Security in Payment Systems
Tampere University of Technology
Master of Science Thesis, 59 pages
May 2016
Master's Degree Programme in Information Technology
Major: Software Engineering
Examiner: Professor Tommi Mikkonen

Keywords: information security, payment system, verification, database, network, server software

The importance of information security is increasing in today's networked world. At the same time, digitalizing the world requires a great deal of new software. Such software often has to be done in a hurry so that the focus on important issues will be more meaningful. Ensuring the safety of financial software is particularly important due to the critical nature of money.

The purpose of this thesis is to offer a guide for payment system projects so that essential and important information security concerns are not forgotten in a busy project schedule. A good base for information security has been ensured for the system if guidelines in this thesis are taken into account. Many aspects in this thesis can be applied for other kind of systems as well. This thesis focuses especially on payment systems because they are critical and they need high availability. Particular focus is on payment authorization service which is highly critical part of a working payment system.

This thesis deals with three different areas of the payment system which together define the information security of the entire system. The first area is the network infrastructure that creates a base for the system. The system communicates with other systems through a network and it also enables decentralization of the system. The second area is the database which is the central point of the system data. Working database guarantees the maintenance and accuracy of the information. The third area is the server software which makes up the functionality of the system. Three important information security concepts confidentiality, integrity and availability are discussed within these areas.

How well the information security concerns were taken care of in this payment system is discussed in the review chapter. There are always many factors that affect information security. Among other things the performance has been able to increase significantly after the payment system was put into production. Information security has also been satisfactory and it has been improved according the recommendations of the information security audits. All this was possible because of a good programming practices as well as comprehensive testing, which allows for making changes easily.

ALKUSANAT

Tämän työn tekemisen apuna oli monia henkilöitä, joita saan kiittää hyvästä avusta ja kärsivällisyydestä. Mielenkiintoisesta aiheesta ja kirjoittamisen tukemisesta kiitän työnantajaani, joka tarjosi hyvät puitteet työn tekemiselle. Erityiskiitos ”Ässä”-tiimille, joka tuki ja kannusti työn saattamisessa valmiiksi. Hyvän aiheen kiitokset kuuluvat myös asiakkaalle, joka tarjosi mahdollisuuden projektille.

Kiitän myös työni tarkastajaa Tommi Mikkosta hyvistä neuvoista ja aikatauluttamisesta. Kysymyksiin sain aina nopeasti vastaukset ja kirjoitusprosessia saatiin hyvin edistettyä.

Suuret kiitokset kuuluvat myös vanhemmilleni ja veljelleni, jotka kannustivat ja auttoivat jaksamaan tämän työn saattamisessa valmiiksi ja erityisesti koko opiskelujeni aikana.

Tampereella, 20.4.2016

Janne Klaper

SISÄLLYSLUETTELO

KUVALUETTELO

TERMIT JA NIIDEN MÄÄRITELMÄT

3G	3G on kolmannen sukupolven matkapuhelinverkkoteknologia. 3G-standardi on määritelty IMT-2000 dokumentissa, mutta muitakin standardeja löytyy. Standardin mukaan piikkinopeus tulisi olla ainakin 2 Mbit/s, mutta monet standardit mahdollistavat jopa yli 14 Mbit/s nopeudet. (Segan, 2015)
4G	4G on neljännen sukupolven matkapuhelinverkkoteknologia, joka mahdollistaa jopa yli 100 Mbit/s yhteydet. 4G-standardeja on useita erilaisia, mutta yleisimmät ovat WiMAX ja LTE (Long Term Evolution). (Segan, 2015)
AAA	AAA (Authentication Authorization and Accounting) on järjestelmä, jolla voidaan seurata käyttäjän tekemisiä sekä rajata pääsy tiettyihin verkkoresursseihin. (Convery, 2007)
Esitystapakerros (OSI)	Esitystapakerros (Presentation layer) on kuudes OSI-mallin kerros. Tämän kerroksen tehtävänä ovat muun muassa huolehtia eri merkistökoodauksien yhteensovittamisesta ja mahdollisesta salaamisesta. (Microsoft, 2014)
Fyysinen kerros (OSI)	OSI-mallin fyysinen kerros (Physical layer) määrittelee millä fyysisellä medialla tiedonsiirto tapahtuu. Yleisimpiä tapoja ovat muun muassa kuparikaapeli ja valokuitu. (Microsoft, 2014)
Istuntokerros (OSI)	OSI-mallin istuntokerros (Session layer) on viides kerros. Istuntokerroksen tehtävänä on huolehtia samassa yhteydessä olevista istunnoista. (Microsoft, 2014)
KISS-periaate	KISS-periaate (Keep It Simple Stupid) on 1900-luvun puolesta välistä tutuksi tullut termi. Termillä tarkoitetaan, että asiat pitää suunnitella yksinkertaisiksi ja turhaa monimutkaisuutta tulisi välttää. (Rich, 1995)
Kuljetuskerros (OSI)	Neljäs kerros OSI-mallissa on kuljetuskerros (Transport layer). Kuljetuskerroksen tarkoitus on huolehtia pakettien saapumisesta määränpäähän ja järjestää ne oikeaa

	järjestykseen. TCP- ja UDP-protokollat toimivat tällä kerroksella. (Microsoft, 2014)
Kytkin	Kytkin yhdistää laitteita toisiinsa pakettikytkentäisessä verkossa. Kytkimeen liitetyt laitteet voivat liikennöidä keskenään. Muualle verkkoon liikennöinti vaatii reitittimen, joka osaa reitittää liikenteen oikeaan paikkaan. Kytkin toimii OSI-mallin siirtokerroksessa. (Cisco, 2016)
OSI-malli	OSI-malli kuvaa tiedonsiirtoprotokollat seitsemään eri kerrokseen. Nämä kerrokset ovat alhaalta ylöspäin seuraavat: fyysinen-, siirto-, verkko-, kuljetus-, istunto-, esitys- ja sovelluskerros. Nämä kerrokset on kuvattu tarkemmin erikseen termien määrittelyssä. (Microsoft, 2014)
RAID	RAID (Redundant Array of Independent Disks) on levyjärjestelmä, jonka avulla voidaan parantaa levyjen suorituskykyä, varmistaa niiden toiminta vikatilanteissa tai molempia. Yksi yleisimmistä RAID-tiloista on RAID 1, joka tarkoittaa kahden levyn peilaamista keskenään. (Habas & Sieber, 2006)
Reititin	Reititin on verkonlaite, joka yhdistää muita verkkoja toisiinsa reitittämällä liikenteen oikeaan paikkaan. Reititys toimii laitteiden IP-osoitteiden perusteella. Reitittimellä on reititystaulu, jonka avulla se osaa liikennöidä oikeaan suuntaan paketin kohde IP:tä tarkastelemalla. Reititin toimii OSI-mallin verkkokerroksessa. (Cisco, 2016)
REST	REST (REpresentational State Transfer) on http protokollaan perustuva rajapintamekanismi. Tällä periaatteella voidaan rakentaa rajapintoja, joita kutsutaan eri URL-poluista käyttäen http-protokollan eri metodeja. (OWASP, 2015)
Siirtokerros (OSI)	Siirtokerros (Data link layer) on OSI-mallin toinen kerros. Tämän kerroksen tehtävänä on siirtää tietoa kahden pisteen välillä ja korjata mahdolliset fyysisen kerroksen aiheuttamat virheet. Kytkimet toimivat tällä kerroksella. Tämän kerroksen protokollia on muun muassa Ethernet. (Microsoft, 2014)

SLA	SLA (Service-level agreement) on asiakkaan ja palveluntarjoajan välinen sopimus, joka kertoo palvelun tason, jonka palveluntarjoaja tarjoaa asiakkaalle. (Service Level Agreement Zone, 2015)
Sovelluskerros (OSI)	Ylin kerros OSI-mallissa on sovelluskerros (Application layer). Tämän kerroksen protokollat ovat sovelluksen käyttämiä protokollia. Esimerkiksi HTTP ja FTP ovat sovelluskerroksen protokollia. (Microsoft, 2014)
SQL	SQL (Structured Query Language) on relaatiotietokantojen yleisin kieli. Kielen avulla voidaan suorittaa erialisia operaatioita tietokannassa, kuten tietojen hakua ja tallentamista. (Microsoft, 2016)
T-SQL	T-SQL (Transact-SQL) on SQL kielen versio, jota käytetään Microsoftin SQL -palvelimilla. (Microsoft, 2016)
Verkkokerros (OSI)	Verkkokerroksen (Network layer) tehtävänä on ohjata paketit oikealle vastaanottajalle. Tämä OSI-mallin kolmaskerros tarjoaa päästä päähän pakettien välityksen. Paketit ohjautuvat oikeaan kohteeseen reitityksen avulla, joka on reitittimien tehtävänä. Yleisin protokolla tällä kerroksella on IP-protokolla. (Microsoft, 2014)
VPN	VPN (Virtual private network) on yksityisten verkkojen yhdistämiseen tarkoitettu menetelmä. VPN:n avulla voidaan esimerkiksi yhdistää yksityiset verkon julkisen Internetin yli niin, että liikenne on salattu tällä välillä. VPN menetelmä tukee useita eri protokollia, jolla tämä voidaan toteuttaa kuten IPSec ja L2TP. (Microsoft, 2001)

1. JOHDANTO

Erilaisia maksujärjestelmiä löytyy maailmasta paljon. Osa maksujärjestelmistä toimii käteisellä, toiset maksukorteilla ja jotkut suoraan Internetissä ilman mitään erillistä fyysistä välinettä. Kaikille niille on yhteistä rahan liikkuminen maksujärjestelmän sisällä sekä järjestelmien välillä. Tällaisia järjestelmiä toteuttaessa pitää olla erityisen tarkka rahansiirroissa, ettei rahaa häviä tai siirry väärään paikkaan. Virtuaaliset maksujärjestelmät ovat yleistyneet entisestään, koska niiden avulla saavutetaan tarvittava joustavuus eri maksutapojen tullessa käyttöön. Tavallisen käteisen rahan käyttö on vähenemässä, kun uusia helpompia maksuratkaisuja tulee markkinoille (Rommann, 2014). Vanhat järjestelmät ovat monesti kankeita eikä niihin saada helposti tehtyä muuttuvan maailman tarvitsemia muutoksia.

Rahan käsittelyyn tehdyt ohjelmat ovat kriittisiä ja monet nykyisistä pankkijärjestelmistä ovat vanhoja. Vanhat järjestelmät on kuitenkin todettu toimiviksi, eikä niitä ole uskallettu uusia. Pankeilla on myös paljon vanhoja palveluja, joita ei ole haluttu poistaa käytöstä. Jotta uudempia ominaisuuksia on saatu käyttöön, on jouduttu käärimään olemassa olevia rajapintoja uudempien järjestelmien sisään. Käärimisellä vältytään olemassa olevan järjestelmän muuttamiselta. Rajapintojen käärimistä on tehty paljon, jotta saadaan uusia ominaisuuksia käyttöön. Kääriminen tuo omat haasteensa ohjelmistokehitykseen ja se saattaa vaikeuttaa järjestelmän ylläpidettävyyttä sekä suorituskykyä.

Tietoturvallisuuden tärkeys korostuu, kun järjestelmät ovat yhä enemmän yhteydessä Internetiin. Internet mahdollistaa sen, että näihin järjestelmiin on mahdollista päästä käsiksi mistä päin maailmaa tahansa. Tämän takia pääsyä tietoon täytyy pystyä rajoittamaan oikeille henkilöille sekä rajoittaa henkilöiden oikeuksia tietoon. AAA (Morand & Dekok, 2014) (Authentication Authorization and Accounting) on yksi järjestelmä, jonka avulla voidaan tarkastella käyttäjien tekemisiä järjestelmässä.

Tutkittavana järjestelmänä on asiakkaalle toteutettu maksujärjestelmä. Tutkimuskohteena on toteutetun järjestelmän tietoturvallisuus ja sen huomioiminen eri osa-alueilla. Maksujärjestelmän loppukäyttäjinä ovat ostoksia tekevät henkilöt, jotka käyttävät maksamiseen tämän maksujärjestelmän maksutapoja. Loppukäyttäjät voidaan tunnistaa monilla eri tavoilla kuten maksukortti ja PIN-koodi, käyttäjätunnus ja salasana tai mobiilisovellus ja PIN-koodi. Tunnistamisen jälkeen loppukäyttäjä voi hyväksyä maksutapahtuman.

Tässä diplomityössä käydään läpi asioita, joita pitää ottaa huomioon tietoturvan kannalta, kun suunnitellaan maksujärjestelmää. Tietoturvaa käsitellään eri osa-alueilla, ja tämä työ antaa hyvän pohjan maksujärjestelmäprojektin projektipäällikölle. Näiden asioiden avulla saadaan hyvä kokonaisuus asioista, joita täytyy huomioida tietoturvallisuuden näkökulmasta. Esimerkkinä työssä käytetään maksujärjestelmäprojektia. Työn arvioinnissa käydään läpi, kuinka hyvin esimerkin maksujärjestelmä sai toteutettua tietoturvallisuuden.

Tutkittavassa maksujärjestelmäprojektissa olin mukana ohjelmistokehittäjänä. Tehtäviini kuuluuivat muun muassa backend-järjestelmien suunnittelu ja toteutus ja erityisesti rajapintojen tekeminen muihin järjestelmiin sekä varmennuksen suorituskyvyn parannukset. Erityisesti varmennus saatiin paljon parannettua tuotantoon viennin jälkeen, kuten arvioinnissa voidaan huomata. Tuotantoon viennin jälkeen työtehtäviini kuului myös järjestelmän ylläpitäminen.

Tämän työn lukijan oletetaan ymmärtävän ohjelmistojen tuottamisen perusasiat. Perusasioihin kuuluu muun muassa verkkopohjaisten järjestelmien korkeatasoisen arkkitehtuurin sekä rajapintojen ymmärtäminen. Korkeatasoisella arkkitehtuurilla tarkoitetaan verkkopohjaisen järjestelmän komponenttien liittymistä toisiinsa, esimerkiksi front- ja backendin sekä mahdollisen tietokannan toiminta ja näiden yhdistyminen toisiinsa. Tutkittavana oleva järjestelmä on laaja kokonaisuus eikä se rajoitu vain verkko-ohjelmien ja rajapintojen tekemiseen, vaan järjestelmän keskeisin osa on tietokanta. Tietokannan perustietämys on myös hyväksi. Tämä työ on tarkoitettu esimerkiksi vastaavan järjestelmän suunnittelijan, arkkitehdin tai projektipäällikön tueksi. Työn tarkoituksena on tarjota dokumentti tietoturvallisuuteen liittyvistä asioista, jotta tietoturvallisuus ei jäisi huomioimatta.

Varmennuksella tarkoitetaan tapahtumaa, jossa tarkastetaan, voiko loppukäyttäjä ostaa vai ei. Varmennus on monimutkainen prosessi ja se selitetään tarkemmin luvussa 2. Varmennuksen lisäksi luvussa 2 selvennetään myös tarkemmin tutkimuskohdetta. Luvussa on hahmotelma järjestelmän komponenteista, josta nähdään miten varmennuspalvelu sijoittuu järjestelmään. Luvussa käydään lisäksi tietoturvallisuuden perusperiaatteet eli miten tietoturvallisuutta voidaan mitata.

Luvussa 3 käsitellään verkon suojausta ja varmentamista eli verkon toimivuuden takaamista. Tämän luvun asiat ovat melko yleisiä, ja monia niistä voi hyödyntää myös muissa projekteissa. Verkon suojaus ja sen suunnittelu on erittäin tärkeää, koska se suojaa järjestelmiä ulkopuolelta tulevia uhkia vastaan. Suurimpia uhkia tällaiselle maksujärjestelmälle ovat palvelunestohyökkäykset sekä hyökkääjän tunkeutuminen järjestelmän sisäverkkoon. Ilman palomuureja ja verkkovyöhykkeiden suunnittelua monet järjestelmät olisivat haavoittuvampia verkosta tulevia uhkia vastaan. Näiden lisäksi pitää myös huomioida verkon varmennus eli huolehtia verkon toiminnasta, vaikka osa laitteista rikkoutuisi tai verkkoyhteys katkeaisi.

Verkko-osuuden jälkeen siirrytään tutkimaan tietokannan turvaamista luvussa 4. Tietokanta on kriittinen osa järjestelmää, koska se sisältää kaikki järjestelmän tiedot. Tietokannassa on tallennettuna kaikki loppukäyttäjien tiedot, kuten tehdyt ostot ja maksetut rahat. Tietokannan turvaamisessa täytyy olla erityisen huolellinen sen sisältämien arkaluontoisten tietojen takia. Tärkeimpiin asioihin kuuluu pääsynvalvonta ja -hallinta sekä hyvä varmuuskopioiden ja lokien luonti. Lokien avulla voidaan selvittää, kuka on muuttanut tietoja ja palauttaa tiedot takaisin oikeiksi. Lisäksi luvattomat käytöt tulisi pystyä huomaamaan sekä estämään.

Palvelinohjelmisto ottaa vastaan verkosta tulevat kyselyt ja käyttää tietokannan tietoja vastauksen antamiseen. Esimerkiksi varmennuspalvelu ottaa vastaan verkosta varmennuskyselyitä ja tarkastaa tietokannan tietojen avulla, annetaanko hyväksytty vai hylätty vastaus. Palvelinohjelmistoja on toteutetussa järjestelmässä useita. Varmennuspalvelun lisäksi järjestelmään kuuluu muun muassa käyttöliittymän tarjoavat palvelinohjelmistot. Palvelinohjelmiston luotettava toiminta on myös tärkeää, ja se, miten tämä on huomioitu, käsitellään luvussa 5. Tässä luvussa käydään läpi niin ohjelmointivaiheessa huomioitavia asioita kuin myös se, miten toiminta on varmistettu, kun ohjelma on tuotannossa.

Järjestelmän arviointia tarkastellaan luvussa 6. Arvioinnissa käydään läpi, miten järjestelmän tietoturvallisuus on saatu toteutettua, ja miten sen toiminta on lähtenyt käyntiin tuotantokäytössä. Arvioinnissa käydään lisäksi läpi sitä, miten helposti tähän varmennukseen voidaan tehdä muutoksia, ja varmistaa sen oikea toiminnallisuus muutosten jälkeen.

Viimeisessä luvussa on diplomityön yhteenveto. Yhteenvedossa tarkastellaan, miten hyvin rakennettu maksujärjestelmä on saavuttanut riittävän tietoturvallisuuden, sekä miten tässä diplomityössä on saatu kuvattua tietoturvallisuuden huomioimisen kannalta tärkeimmät asiat.

2. SOVELLUSKOHDE

Sovelluskohteena on maksujärjestelmä, joka toteutettiin räätälöitynä ratkaisuna. Järjestelmän tarkoituksena oli korvata edellinen järjestelmä sekä jatkaa sen kehitystä asiakkaan toiveiden mukaan. Toteutetussa järjestelmässä asiakas voi muuttaa sen toimintoja paljon vapaammin. Maksujärjestelmäprojektin aikataulu oli kiireellinen, jonka vuoksi oli tärkeätä varmistaa riittävä tietoturvallisuustaso. Järjestelmä saatiin määräajassa vietyä tuotantoon. Tämän jälkeen se on saanut useita päivityksiä, jossa sen ominaisuuksia on parannettu ja uusia toimintoja lisätty.

Maksujärjestelmän tietoturvallisuutta tarkastellaan monesta eri näkökulmasta. Varmennus on kuitenkin järjestelmän keskeisin ja kriittisin osa, joka mahdollistaa järjestelmän toiminnan. Varmennuksen tekemiseen on myös panostettu paljon, jotta sen luotettavuus olisi halutulla tasolla. Sen toimintaa tarkkaillaan, jotta mahdolliset virhetilanteet saadaan nopeasti korjattua. Muita kriittisiä osia ovat muun muassa rajapinnat, jotka näkyvät Internetiin sekä järjestelmän sisäinen toiminta.

Tehdyn järjestelmän suunnittelussa ja toteutuksessa on pitänyt huomioida lisäksi järjestelmän suorituskyky ja mahdolliset tarpeet loppukäyttäjämäärän kasvaessa. Loppukäyttäjämäärän kasvaessa järjestelmään tulevien varmennusten määrä kasvaa ja vaikuttaa tällöin myös varmennuspalvelun toimintaan ja saatavuuteen. Järjestelmä on suunniteltu palvelemaan vähintään yli sataatuhatta loppukäyttäjää (Maksujärjestelmäprojektiryhmä, 2015). Nykyisistä käyttäjämääristä järjestelmä on suoriutunut hyvin. Varmennusaikoja tarkkaillaan säännöllisesti, jotta mahdollisia parannuksia voidaan tehdä tulevan kasvun myötä. Varmennusajalla tarkoitetaan järjestelmän käyttämää aikaa yhteen varmennuskyselyyn.

Tässä luvussa käydään läpi sovelluskohteen kannalta tärkeät asiat. Ensimmäisessä kohdassa 2.1 käsitellään maksujärjestelmää ja mitä sillä tarkoitetaan. Kohdassa 2.2 käydään läpi varmennusta, joka on maksujärjestelmän keskeinen osa. Kohdassa 2.3 selvennetään tietoturvan käsitettä ja mitä sillä oikeastaan tarkoitetaan. Lopuksi, kohdassa 2.4, määritellään SLA:n merkitys. SLA-tasolla on merkittävä osa tietoturvallisuuden saatavuuden kanssa, koska sen avulla saadaan sopimuksiin tarvittava palvelun saatavuusaika.

2.1 Maksujärjestelmä

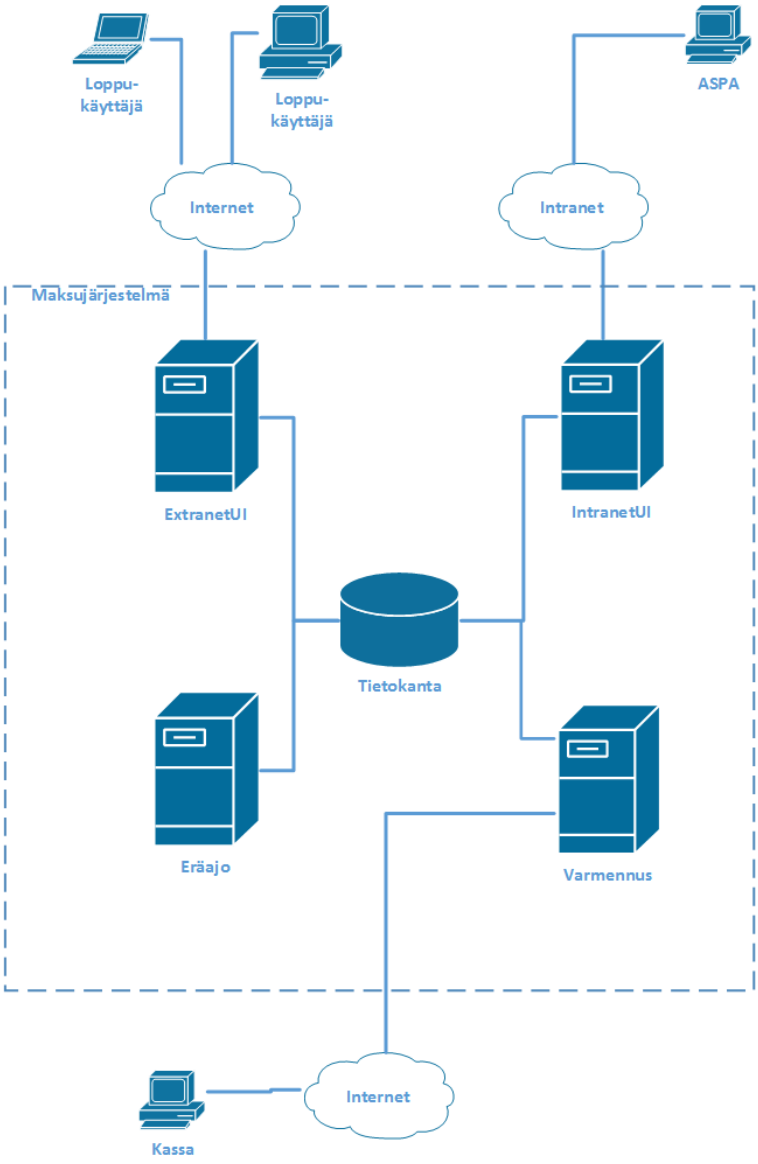
Maksujärjestelmällä tarkoitetaan järjestelmää, joka mahdollistaa ostoksien tekemisen jollakin maksuvälineellä, kuten maksukortilla, mobiililaitteella tai muuten tunnistamalla

asiakas. Laskut ostoksista lähetetään takautuvasti. Toteutetulla maksujärjestelmällä loppukäyttäjät voivat ostaa aina luottorajaan asti. Toinen mahdollisuus on maksaa järjestelmään ennakkoon jokin summa, jolloin palvelu toimii ennakkomaksujärjestelmänä.

Tutkittavana oleva maksujärjestelmä huolehtii tietojen hallinnasta järjestelmän sisällä. Se on myös integroitu moniin muihin järjestelmiin. Esimerkiksi ostojen tarkemmat tiedot ja käyttäjien suoritukset tulevat järjestelmään ulkopuolelta. Järjestelmästä ulospäin tuotetaan muun muassa laskut ja muihin järjestelmiin vietävät aineistot. Tiedonsiirto eri järjestelmien välillä tapahtuu tiedostojen välityksellä ja joihinkin järjestelmiin on toteutettu lisäksi rajapinta kommunikointiin. (Maksujärjestelmäprojektiryhmä, 2015)

Tarkasteltavana olevassa maksujärjestelmässä on pääasiassa neljä osaa, jotka tarjoavat järjestelmän ulkoiset ja sisäiset liitynnät. Selainkäyttöliittymiä on järjestelmässä kaksi, joista toinen on asiakaspalvelulle (ASPA) ja toinen on asiakkaille tarjottava käyttöliittymä. Asiakaspalvelun käyttöön tehty käyttöliittymä on nimetty IntranetUI:ksi ja loppukäyttäjien käyttöliittymä ExtranetUI:ksi. Eräajoihin perustuvan palvelun avulla voidaan tuoda tietoa järjestelmään ja viedä sieltä myös ulos. Tärkeimpänä palveluna on tietenkin varmennuspalvelu, jonka kautta kaikki varmennuskyselyt tulevat järjestelmään. Näitä kaikkia palveluja yhdistää tietokanta, josta tarvittavat tiedot saadaan haettua. Kuva 1 havainnollistaa tämän järjestelmän yksinkertaisesti. Tämä kuva on kuitenkin pelkistetty todellisesta toteutuksesta ja todellisuudessa järjestelmä pitää sisällään myös muita osia. Nämä muut osat sitovat ja tukevat järjestelmän toimintaa ja mahdollistavat turvallisen tietokannan käytön. (Maksujärjestelmäprojektiryhmä, 2015)

Kuvassa 1 on toteutetun maksujärjestelmän hyvin yksinkertainen kuvaus. Kuvasta nähdään kuitenkin järjestelmän pääkomponentit. Järjestelmä koostuu ExtranetUI-, IntranetUI-, eräajo- ja varmennuspalveluista. ExtranetUI:lla tarkoitetaan järjestelmän loppukäyttäjille tarjottavaa käyttöliittymää, johon otetaan yhteyksiä ulkoverkon kautta. IntranetUI on asiakaspalvelun käyttöliittymä järjestelmään ja sen käyttäjät tulevat yrityksen sisäverkosta. Eräajopalvelu on järjestelmän sisäinen palvelu, jolla ei ole yhteyksiä ulkomaailmaan. Eräajopalvelua käyttävät pääasiassa vain järjestelmän ylläpitäjät. Varmennuspalvelu on yhteydessä vain maksuliikenteen reititysverkkoon, johon on avattu VPN (Virtual Private Network) yhteys. Tämän yhteyden avulla voidaan varmistaa, että vain sallittua liikennettä tulee palvelulle. Näitä kaikkia komponentteja yhdistää yhteinen tietokanta, josta kaikki tiedot haetaan tai tallennetaan. (Maksujärjestelmäprojektiryhmä, 2015)



Kuva 1: Toteutetun maksujärjestelmän pääkomponentit.

Asiakaspalvelun käyttöön tehdyllä käyttöliittymällä hallitaan loppukäyttäjien tietoja ja seurataan järjestelmän toimintaa. Käyttöliittymän avulla voidaan tehdä uusien loppukäyttäjien lisäyksiä ja nykyisten tietojen muokkausta, sekä tarkastella erilaisia järjestelmän keräämiä tietoja. Käyttöliittymästä on tehty selkeä, ja sen suunnittelussa on huomioitu myös asiakaspalveluhenkilöstön toiveita. Lisäksi sen suorituskykyä on parannettu koko ajan, jotta asiakaspalvelusta saataisiin tehokasta. Selainkäyttöliittymä mahdollistaa myös sen käyttämisen miltä koneelta tahansa, jos sinne on verkkotasolta annettu pääsy ja henkilöllä on tarvittavat oikeudet. (Maksujärjestelmäprojektiryhmä, 2015)

Loppukäyttäjän selainkäyttöliittymän, ExtranetUI:n, avulla voi tarkastella tietoja ja luoda raportteja. Tämän avulla loppukäyttäjien on siis mahdollista saada ostotiedot jälkikäteen tai seurata niitä, kun nämä tiedot on tuotu järjestelmään. Käyttöliittymä

mahdollistaan vuorovaikutuksen suoraan asiakaspalvelun kanssa, jolloin loppukäyttäjä voi muun muassa tilata uusia tilejä ja muuttaa käyttäjätietoja. Verkkosivujen selaaminen mobiilipäätteillä on lisääntynyt paljon, joten myös tämän verkkosivun rakentamisessa on huomioitu mobiilipäätteitä ja niiden tuomia rajoituksia. (Maksujärjestelmäprojektiryhmä, 2015)

Eräajopalvelun avulla tuodaan järjestelmään sisään tietoja muista järjestelmistä. Tällaisia tietoja ovat esimerkiksi ostotiedot ja viitesuoritukset, jotta loppukäyttäjät saavat ostotiedot tämän järjestelmän kautta, sekä loppukäyttäjien tekemät suoritukset. Järjestelmästä vietävät tiedot muita järjestelmiä varten tehdään myös eräajopalvelussa. Tällaisia tietoja ovat muun muassa kirjanpitoaineistot, loppukäyttäjille lähtevät laskut sekä tietojen vienti asiakkaan muihin järjestelmiin. Järjestelmä joko lukee annetuista tiedostoista tiedot järjestelmään tai luo tiedostoja muiden järjestelmien ymmärtämässä muodossa. Tällöin järjestelmän ei tarvitse olla suoraan yhteydessä muihin järjestelmiin, koska tiedostojen siirto eri järjestelmien välillä on kolmannen osapuolen ohjelma, jolle on annettu tarvittavat oikeudet. (Maksujärjestelmäprojektiryhmä, 2015)

Tietojen tuonnin ja viennin lisäksi palvelulla luodaan useita erilaisia raportteja, joiden avulla varmistutaan järjestelmän oikeasta toiminnasta sekä täsmätään tietoja muiden järjestelmien tuottamia raportteja vasten. Monet päivittäiset tietojen tuonnit ja viennit on ajastettu tällä palvelulla automaattisesti ajettavaksi, mutta toimintoja voidaan myös ajaa manuaalisesti. Tätä järjestelmää seuraa pääasiassa järjestelmän ylläpito, joka tarkastaa eräajojen läpimenemisen ja ilmoittaa mahdollisista poikkeuksista. (Maksujärjestelmäprojektiryhmä, 2015)

Varmennuspalvelu on järjestelmän toiminnan kannalta yksi tärkeimmistä ja kriittisimmistä palveluista. Varmennuspalvelulla otetaan vastaan järjestelmään tullut varmennuspyyntö ja annetaan joko kielteinen tai myönteinen vastaus. Ennen vastauksen antamista pitää kuitenkin tarkastaa, voiko kyseinen loppukäyttäjä ostaa ja onko hänellä tarpeeksi katetta. Vaikka asia kuulostaa yksinkertaiselta, sen pitää kuitenkin toimia luotettavasti ja tehokkaasti. Liian pitkään kestävä varmennus saattaa aiheuttaa harmia loppukäyttäjälle tai laskuttajalle, joka haluaa maksun nopeasti ostohetkellä. Varmennuspalvelun katkot aiheuttavat loppukäyttäjille haittaa eikä katkojen aikana saa maksua suoritettua. Tämän takia varmennuspalvelun toimivuuteen on panostettu paljon, ja sitä on optimoitu, jotta se olisi mahdollisimman nopea ja toimisi luotettavasti myös virhetilanteissa. (Maksujärjestelmäprojektiryhmä, 2015)

2.2 Varmennus

Nykyisessä toteutuksessa on paljon loppukäyttäjiä, joilla on käytössä paljon erilaisia maksutapoja. Lähes kaikille maksutavoille on yhteistä varmennus, joka tehdään jokaisen maksun yhteydessä. Maksujärjestelmä perustuu siis online-varmennukseen (Maksujärjestelmäprojektiryhmä, 2015). Maksukortit sisältävät tyypillisesti sirun, joka

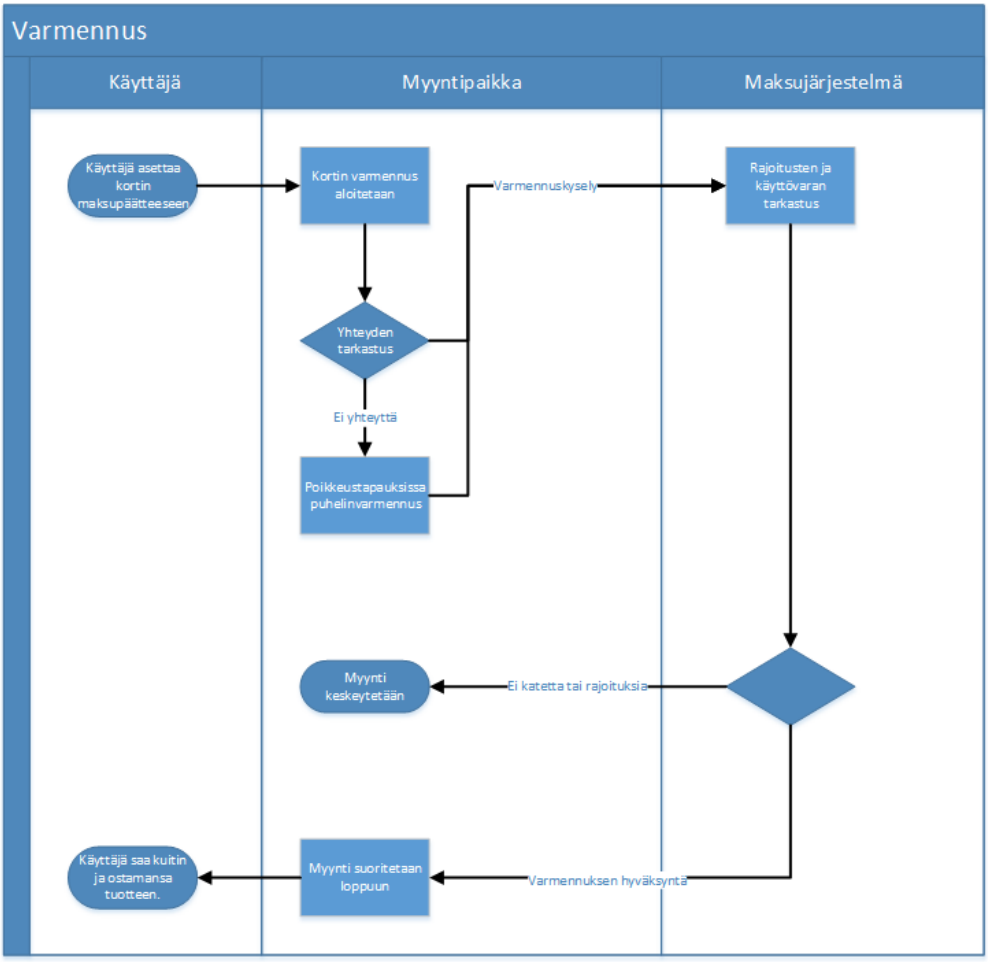
luo turvallisuutta maksamiseen. Tällöin puhutaan EMV-korteista. EMV-korttien ja niiden varmennus on kansainvälisesti standardoitu EMVCo:n toimesta. EMVCo on perustettu 1999 kehittämään ja standardoimaan EMV maksutapahtumien käsittelyä. Myös sirulla voi olla tietoa, joka säätelee missä tapauksessa varmennusta tarvitaan. Alussa siinä olivat mukana Europay, Mastercard ja Visa, joista lyhenne EMV tulee. Nykyään siinä on mukana kuusi yritystä, ja heidän standardointinsa on laajentunut kehityksen myötä muun muassa lähi- ja mobiilimaksamiseen. (EMVCo, 2015)

Varmennuksella tarkoitetaan prosessia, jossa varmistetaan onko käyttäjällä oikeus ostaa annetulla summalla. Maksujärjestelmän varmennuksen toiminta on tehty Finanssialan keskusliiton ylläpitämän EMV-maksupäätejärjestelmädokumentin mukaisesti. Maksupäätteen ja reitittimen välillä käytetään Finanssialan keskusliiton dokumentissa MPJ 02001 kuvattuja sanomia. ISO 8583 -standardin mukainen sanomarakenne on puolestaan käytössä maksujärjestelmän ja kolmannen osapuolen reitittäjän välillä. Tämä standardi määrittelee viestien muodot sekä mahdolliset erityyppiset vastaukset, joita on mahdollista lähettää. (Finanssialan keskusliitto, 2011)

Ennen kuin varmennuskysely tulee maksujärjestelmälle, on tehty jo monia turvatarkastuksia. Varmennusprosessi alkaa, kun käyttäjä on maksamassa ostoksiaan ja syöttää kortin maksupäätteeseen ja näppäilee kortin PIN-koodin. Ensimmäisessä vaiheessa on tietenkin kassan maksupäätteen tekemä tarkistus oikeasta kortin omistajasta kysymällä käyttäjältä PIN-koodi (Personal Identification Number). Hyväksytyn PIN-koodin jälkeen pääte lähettää varmennuskyselyn, joka tulee maksujärjestelmälle kolmannen osapuolen reitittämänä. Maksujärjestelmän varmennuspalvelu vastaanottaa varmennuskyselyn ja tarkastaa loppukäyttäjän tiedoista, voiko kyseinen loppukäyttäjä ostaa annetulla kortilla. Tähän tarkastukseen kuuluu loppukäyttäjän käyttövaran tarkastaminen, jotta loppukäyttäjä ei ylittäisi asetettuja käyttörajoja. Käyttövaran tarkastuksen lisäksi tarkistetaan, että kyseinen kortti on vielä voimassa eikä loppukäyttäjällä ole muitakaan rajoituksia, jotka voisivat estää kortin käyttämisen. Tarkastamisen jälkeen järjestelmä lähettää joko hyväksytyn tai hylätyn vastauksen. Kuvassa 2 on esitetty varmennusprosessin eri vaiheet. (Maksujärjestelmäprojektiryhmä, 2015)

Hyväksytyn varmennuksen jälkeen tehdään loppukäyttäjälle katevaraus tulevasta veloituksesta. Katevarauksen tarkoituksena on pienentää käyttäjän käyttövaraa kunnes oikea veloitustapahtuma on tuotu järjestelmään tai katevaraus peruutetaan. Katevarauksella pyritään pitämään kortin käyttövara oikeana, jotta loppukäyttäjän käyttöraja ei ylittyisi. Katevaraus poistetaan määrätyn ajan kuluttua, jos kyseisen varmennuksen tapahtumaa ei kirjata järjestelmään. Liian lyhyt aika katevarausten sulkemiseen saattaa kuitenkin aiheuttaa käyttörajan ylityksen, jos tapahtumien kirjaus järjestelmään viivästyy. Myös turhan pitkä aika katevarauksen poistumiselle haittaa loppukäyttäjää, koska katevaraus saattaa turhaan pienentää käyttövaraa. Katevaraus, johon ei tule koskaan oikeaa tapahtumaa, on jonkin virhetapahtuman aikaansaannos.

Tällaista katevarausta ei siis pitäisi tapahtua normaalissa toiminnassa. Näiden varalta on kuitenkin hyvä käydä poistamassa vanhoja katevarauksia. (Maksujärjestelmäprojektiryhmä, 2015)



Kuva 2: Varmennuskyselyn korkeantason prosessi. Kuva on järjestelmän määrittelydokumentaatiosta. (Maksujärjestelmäprojektiryhmä, 2015)

Hylätyn varmennuksen tapauksessa lähetetään kielteinen vastaus varmennukseen, jolloin myynti keskeytyy eikä katevarausta tehdä. Hylätty varmennus voi johtua esimerkiksi kortilla olevista rajoituksista, käyttörajan ylityksestä tai kortin voimassaoloajan päättymisestä. Edellä mainituista sekä muista mahdollisista tapahtumista kirjataan järjestelmään merkintä, jotta mahdollisia ongelmatilanteita voitaisiin tutkia. Näiden avulla saadaan myös tietoa esimerkiksi vanhentuneilla korteilla yritetyistä varmennuksista. (Maksujärjestelmäprojektiryhmä, 2015)

Hyväksyttyjä varmennuksia voidaan myös peruuttaa kokonaan tai osittain. Varmennuksen peruuttaminen kokonaan tarkoittaa maksusuorituksen peruuttamista kesken ostotapahtuman. Varmennuksen peruutus osittain tarkoittaa puolestaan sitä, että ensin on tehty isompi varmennus jollain summalla, mutta toteutunut osto onkin pienempi. Tällöin varmennus peruutetaan osittain, jolloin katevarauksen suuruus on vain toteutuneen oston suuruinen. (Maksujärjestelmäprojektiryhmä, 2015)

Varmennuksen tulee olla nopea prosessi, jotta asiakkaat eivät joutuisi odottamaan ostoa tehdessään. Samalla se pitäisi kuitenkin olla tarkka siitä, että käyttörajoja ei ylitetä. Jotta käyttövaran laskenta saataisiin mahdollisimman tarkaksi, laskenta pitää tehdä reaaliaikaisesti. Tämä saattaa kuitenkin olla hidasta, jos laskettavaa on paljon. Toinen tapa olisi laskea käyttövara etukäteen, mutta tämä saattaisi olla hieman jäljessä varsinkin, jos ostoksia tehdään useita peräkkäin. Näitä kahta tapaa voidaan myös yhdistää ja laskea esimerkiksi harvemmin muuttuvat asiat valmiiksi. Tällöin valmiiksi laskettuun summaan pitäisi laskea mukaan vain useammin muuttuvat summat. Varmennusaikoja tulee tästä syystä seurata säännöllisesti, jotta mahdollisia korjaustoimia voidaan tehdä. Loppuasiakkaan kokemus on kuitenkin merkittävässä osassa siinä, haluaako hän käyttää tätä maksutapaa tehdessään ostoksia. (Maksujärjestelmäprojektiryhmä, 2015)

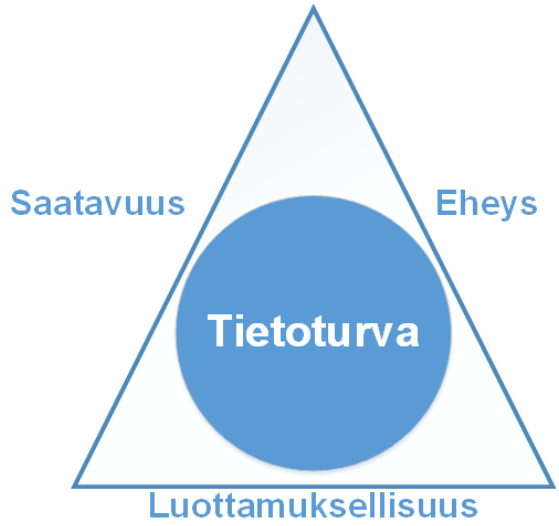
Rakennettu maksujärjestelmä kattaa vain asiakkaan palvelimella toimivan järjestelmän. Kassapäätteet ovat kolmannen osapuolen tarjoama palvelu, ja sieltä tuleva liikenne ohjataan maksujärjestelmään. Varmennusyhteys on TCP/IP-yhteys, joka on salattu varmenteilla (Finanssialan keskusliitto, 2011). (Maksujärjestelmäprojektiryhmä, 2015)

2.3 Tietoturva

Tietoturvan merkitys on kasvanut merkittävästi, ja sen osuus tulee kasvamaan entistä suuremmaksi tulevaisuudessa. Tietoturvallisuutta voidaan arvioida monella tapaa, mutta ISO 27000[1] – sarjan ja NIST[2]:n (National Institute of Standards and Technology) tuottamat standardit antavat hyvän pohjan arviointiin. Myös kotimaiset VAHTI[3]-ohjeet ovat keskeisessä osassa Suomessa sekä tämän järjestelmän arvioinnissa. Tietoturvallisuuden huomioiminen on kuitenkin aina tasapainottelua tietoturvallisuuden ja käytettävyyden välillä. Kaikkea ei ole aina edes mahdollista huomioida. Riskien kartoittamisella saadaan hyvä kuva mahdollisista riskeistä, jotka kohdistuvat järjestelmää kohtaan. Kun mahdollisille riskeille on saatu arvioitua niiden todennäköisyydet, voidaan suunnitella, miten tietoturva toteutetaan järjestelmälle. Kuvassa 3 on tietoturvallisuuden kolme osa-aluetta, jotka ovat luottamuksellisuus, eheys ja saatavuus. (National Institute of Standards and Technology, 2012)

Luottamuksellisuudella tarkoitetaan tiedon luovuttamista vain oikeille henkilöille, joilla on niihin oikeus. Tämä tarkoittaa usein pääsynhallintaa, jolloin tarkastetaan onko henkilöllä tarvittavat oikeudet tietoon. (National Institute of Standards and Technology, 2012)

[1] http://www.27000.org/
[2] http://www.nist.gov/
[3] https://www.vahtiohje.fi/



Kuva 3: Tietoturvallisuuden osa-alueet.

Eheydellä varmistetaan, ettei tieto muutu ilman, että sitä huomattaisiin. Tällöin varmistetaan, että haluttu tieto on juuri kuten pitää tai sen muuttuminen huomataan, jolloin voidaan tehdä tarvittavat toimenpiteet. (National Institute of Standards and Technology, 2012) Yksi tapa varmistaa tiedon eheys on käyttää salausmenetelmiä. Eräs yleisimmistä salausmenetelmistä on epäsymmetrisen salauksen käyttö tiedon allekirjoittamiseen yksityisellä avaimella, jonka tiedon käyttäjä voi purkaa julkisella avaimella. Esimerkiksi suosittu sähköpostin salaus- ja allekirjoitusprotokolla PGP[4] toimii tällä periaatteella.

Saatavuudella puolestaan tarkoitetaan, että haluttu tieto on aina saatavilla (National Institute of Standards and Technology, 2012). Tämän varmistaminen vaatii usein paljon työtä varsinkin, jos tietoon pitäisi päästä käsiksi Internetistä. Tähän ei siis riitä, että järjestelmä on saatavilla, vaan myös verkkoinfrastruktuurin pitäisi toimia erilaisissa vikatilanteissa, kuten sähkökatkoissa. Koska saatavuuden varmistamiseen vaikuttaa paljon erilaiset fyysiset riskit, on tämän takaaminen kallista. Riskienhallinnan avulla voidaan määrittää todennäköisimmät tilanteet ja suojautua niitä vastaan. Järjestelmissä on usein määritetty niiden SLA-tasot (Service-level agreement).

Järjestelmän keräämät lokit ovat tärkeitä tietoturvallisuuden kannalta. Auditointilokien tarkoituksena on kerätä tietoa, kuka on muuttanut järjestelmässä tietoa ja miten sitä on muutettu. Näiden tietojen avulla on mahdollista selvittää tapahtumat jälkikäteen. Jotta järjestelmän tietoturvallisuus voitaisiin varmistaa, voidaan järjestelmälle tehdä tietoturva-auditointi. Auditointi on yleensä kolmannen osapuolen tekemä, jotta arviointi olisi mahdollisimman puolueeton.

[4] http://www.openpgp.org/

2.4 SLA – Service Level Agreement

SLA eli palvelutasosopimus määrittää palvelulle vaatimustason. SLA-taso kuvaa kuinka monta prosenttia ajasta palvelun pitäisi olla saatavilla. Palvelun on siis toimittava SLA-sopimuksen määrittämän ajan ja sen alittamisesta seuraa usein sanktioita. Tietoturvallisuuden saatavuus perustuukin hyvin paljon palvelulle määritettyyn SLA-tasoon. Tarkasteltavana oleva järjestelmän eri osa-alueet sisältävät omat SLA-sopimuksensa, joiden puitteissa niiden on toimittava. Verkolle, tietokannalle sekä itse palvelulle on omat SLA-sopimukset, koska ne ovat eri organisaatioiden toimittamia. Tämä puolestaan tarkoittaa, että koko järjestelmän SLA määräytyy näiden eri osa-alueiden yhteen kerrottujen SLA-tasojen mukaan. SLA:t voidaan laskea näin, kun järjestelmät ovat toisistaan riippuvaisia. Tämä yhdistetyn SLA:n laskeminen on esitetty kaavassa 1.

Kaava 1: Eri SLA-tasojen yhteen laskeminen.

$$SLA = SLA_1 * SLA_2 * SLA_3 * \dots * SLA_n$$

Käyttämällä kaavaa 1 voidaan laskea kolmen komponentin yhdistetty SLA, joka on esitetty kaavassa 2. Tässä esimerkissä on otettu verkon, tietokannan sekä palvelun SLA-tasot. Yhteiseksi SLA-tasoksi saadaan 94,1 %, joka on huomattavasti alhaisempi, kuin yhdenkään komponentin oma SLA.

Kaava 2: Yhdistetyn SLA:n esimerkki.

$$SLA = SLA_{verkko} * SLA_{tietokanta} * SLA_{palvelu} = 99\% * 98\% * 97\% = 94{,}1\%$$

SLA-tasot ovat yleensä hyvin korkealla ja monesti käytetään yli 99 % SLA-tasoja. Esimerkiksi 99,9 % SLA tarkoittaa, että vuodessa palvelu voi olla alhaalla 8,76 tuntia. SLA-tason ollessa 99,99 % saa palvelu olla alhaalla vain 52,6 minuuttia. Tämä on aika suuri ero varsinkin kun kyseessä ovat kriittiset järjestelmät, joiden pitäisi olla saatavilla lähes koko ajan. Näitä eri palvelutasoja on esitetty taulukossa 1, jossa on ilmoitettu tasojen maksimikatkoajat.

Taulukko 1: Katkojen kesto eri saatavuusprosenteilla.

Saatavuus	Vuodessa	Kuukaudessa	Viikossa	Päivässä
95%	18,25 d	1,5 d	8,4 h	1,2 h
99%	3,65 d	7,2 h	1,68 h	14,4 m
99,5%	1,83 d	3,6 h	50,4 m	7,2 m
99,9%	8,76 h	43,2 m	10,1 m	1,44 m
99,99%	52,6 m	4,32 m	1,01 m	8,64 s
99,999%	5,26 m	25,9 s	6,05 s	0,86 s

Kuten taulukosta 1 nähdään, 99,999% saatavuus ei jätä juurikaan varaa järjestelmän toimimattomuudelle. Monesti järjestelmien SLA:t ovat yli 99%, mutta prosentin pudotus mahdollistaa jo aika pitkän katkon. Tämä kannattaa huomioida, kun tekee sopimuksia ja pitäisi turvata saatavuutta. Sopimuksissa järjestelmän SLA:n rikkomisesta on yleensä määrätty jonkinlainen sakko, jonka järjestelmän toimittajan tulee korvata.

3. VERKKOARKKITEHTUURI

Nykyään lähes kaikki järjestelmät ovat kiinni Internetissä. Ulkopuolisen hyökkääjän on mahdollista ottaa yhteyksiä tällaisiin järjestelmiin. Ensimmäisenä on kuitenkin vastassa tunkeuduttavan järjestelmän oma verkko ennen kuin päästään mahdolliseen kohdekoneeseen käsiksi. Verkon suunnittelu ja suojaus on tärkeässä asemassa, koska sen avulla saadaan rajoitettua yhteyksien ottamista vain tiettyihin resursseihin. Verkon suojauksella voidaan myös havaita ja estää tunnettuja hyökkäyksiä.

Verkon suojauksen lisäksi tulee varmistaa verkon toimivuus vikatilanteissa. Jotta verkossa ei olisi yhtä heikkoa lenkkiä, tulee kaikki laitteet kahdentaa. Tämä tarkoittaa myös verkkoyhteyksien kahdentamista. Verkon toimivuus on kuitenkin hyvin kriittinen osa järjestelmää. Toimivuuden varmistamisella saadaan toteutettua tietoturvallisuuden saatavuutta. Saatavuus on liiketoiminnallisesti tärkeätä myös toteutetussa maksujärjestelmässä, jonka tulisi pystyä vastaanottamaan varmennuksia jatkuvasti.

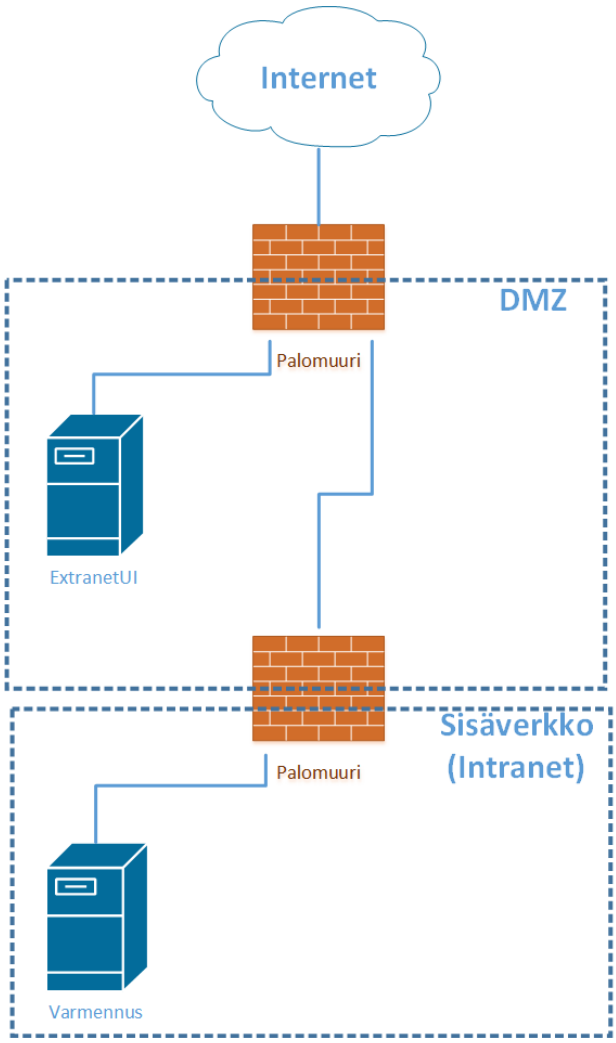
Asioita tarkastellaan tässä luvussa lähinnä yleisellä tasolla. Periaatteet ovat siis yleisiä monille muillekin järjestelmille kuin tarkasteltavana olevalle maksujärjestelmälle ja niitä voidaan hyödyntää myös muiden verkkojen suunnittelussa. Toteutetun maksujärjestelmän verkkosuunnittelu on monelta osin abstrahoitu järjestelmän toteuttavalle taholle, koska sen ylläpitämisestä vastaa asiakkaan palveluntarjoaja. Kuva 1 havainnollistaa erilaisten verkkojen olevan maksujärjestelmän oleellinen osa. Maksujärjestelmän tarvitsee sallia myös ulkopuoliset yhteydet, jotta kommunikointi muiden järjestelmien välillä olisi mahdollista.

Kommunikointia varten tarvitaan verkko, jonka suojauksesta ja varmistuksesta kerrotaan tässä luvussa. Ensimmäisessä kohdassa 3.1 käsitellään verkon suojausta, joka tärkein osa verkon tietoturvassa. Verkon varmistusta käsitellään kohdassa 3.2. Varmistamisella pyritään rakentamaan verkko, joka toimii myös erilaisissa häiriötilanteissa. Varmistuksen tehtävänä on taata saatavuus verkkotasolla. Lopuksi kohdassa 3.3 käsitellään eri verkkojen yhdistämistä turvaamattoman Internetin yli.

3.1 Verkon suojaus

Verkon suojauksella tavoitellaan eheyttä ja luottamuksellisuutta. Peruslähtökohta suojaukseen on verkon hyvä suunnittelu eri osa-alueisiin. Osa-alueet jaotellaan yleensä eri turvaluokan osiin. Tällöin eri turvaluokan tieto voidaan säilöä eri puolille verkkoa riippuen, miten paljon suojausta tieto vaatii tai mistä sinne saa ottaa yhteyksiä. Verkon osituksella saadaan rajoitettua tiedon saatavuutta vain haluttuihin paikkoihin. Tämän

tapaiset verkot on rakennettu usein kerrosmaisesti. Kerrosmaisuudessa ulkoverkon ja sisimmän turvallisimman verkon välillä on useita palomuureja valvomassa, että vain sallittu liikenne menee läpi. Kuva 4 havainnollistaa hyvin yleisen tavan verkkojen osittamiseen. Kuvasta nähdään myös palomuurin sijoitus verkon suojauksessa. (Valtiohallinnon tietoturvallisuuden johtoryhmä, 2010)



Kuva 4: Verkkoalueiden yleinen periaate.

Kuvassa 4 on perinteinen tapa osittaa verkkoja. DMZ (Demilitarized zone) on verkon heikoimmin suojattu alue. Heikko suojaus ei kuitenkaan tarkoita, että sinne olisi helppo murtautua. DMZ-alue on yleensä se verkkoalue, jonne sallitaan ulkopuoliset yhteydet. Ulkopuolisten yhteyksien sallimisella halutaan yleensä tarjota joitain palveluja ulkomaailmaan, jolloin sinne pitää päästä ulkopuolelta ottamaan yhteyksiä. Tällaisia palveluja on esimerkiksi käyttäjille tarjotut www-palvelut. Tutkittavan järjestelmän ExtranetUI on tällainen palvelu (Maksujärjestelmäprojektiryhmä, 2015). (Valtiohallinnon tietoturvallisuuden johtoryhmä, 2010)

Edellä esitellyn verkon osituksen lisäksi on mahdollista osittaa myös sisäverkko erilaisiin alueisiin suojattavan tietoturvatarpeen mukaan. Esimerkiksi kohdejärjestelmän

tietokanta voisi olla turvallisessa verkkosegmentissä sisäverkon sisällä. Näin saataisiin tietokannan tiedot turvallisempaan paikkaan ja lisää suojaa mahdollisia murtautumisia vastaan. Kuten kuvasta voidaan havaita, on palomuuri tärkeässä osassa verkkojen suojauksessa. Sen tärkeimpänä tehtävänä on tarkastella verkkoliikennettä ja sallia oikeanlainen liikenne ja estää kaikki muu. (Valtiohallinnon tietoturvallisuuden johtoryhmä, 2010)

Verkon suojaaminen pelkästään palomuurilla ei ole riittävä toimenpide. Jotta mahdolliset tunkeutumiset voitaisiin havaita ja mahdollisesti estää, tarvitaan muita järjestelmiä tämän saavuttamiseksi. Tällaisia järjestelmiä ovat esimerkiksi tunkeutumisen tunnistusjärjestelmä IDS (Intrusion Detection System) sekä tunkeutumisen estojärjestelmä IPS (Intrusion Prevention System). Näiden järjestelmien avulla saadaan mahdolliset tunkeutumiset havaittua ajoissa, jotta niihin pystytään reagoimaan. Tällaiset järjestelmät toimivat yleensä hyökkäyksistä kerättyjen sormenjälkien perusteella. Järjestelmät tarkastelevat liikennettä ja yrittävät löytää tiedossa olevien sormenjälkien perusteella merkkejä mahdollisesta hyökkäyksestä. Koska kehittyneemmät hyökkäykset ja virukset osaavat muokata itseään jatkuvasti, eivät tällaiset järjestelmät välttämättä havaitse uusia hyökkäyksiä ja viruksia. Uudet järjestelmät käyttävätkin kehittyneitä algoritmeja eivätkä nojaudu enää sormenjälkien etsimiseen. Tällöin myös uudet haittaohjelmat on mahdollista havaita, vaikka ne muuttuisivatkin. (Valtiohallinnon tietoturvallisuuden johtoryhmä, 2010)

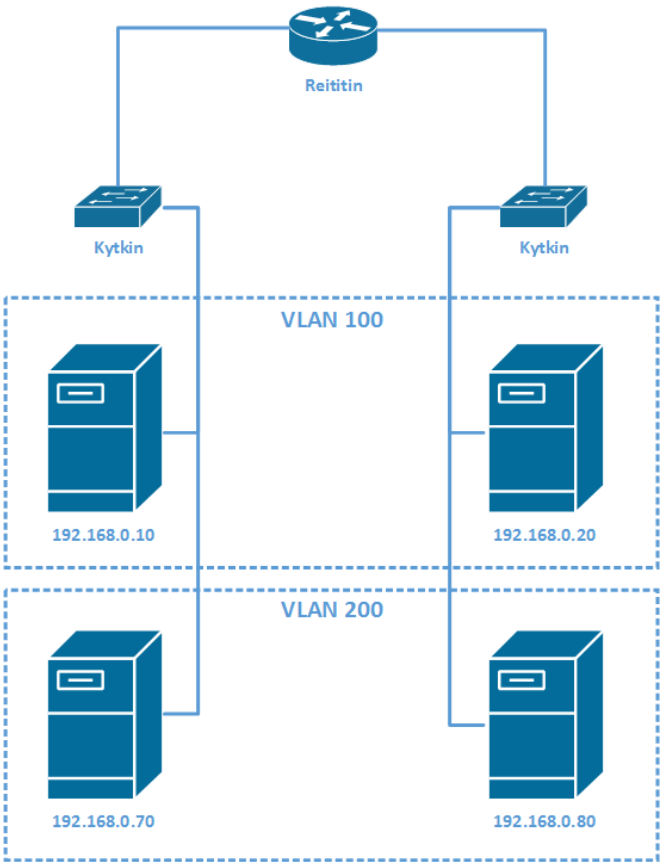
3.1.1 Verkkoalueet

Verkkoalueiden määrittämisessä on pääosassa palomuuri. Palomuurilla saadaan rajattua liikennettä oikeille tahoille sekä suodattamaan ei haluttu liikenne pois. Palomuurit sijoitetaankin verkkoalueiden reunalle varmistamaan turvallisuus. Verkkoalueet saadaan luotua reitittimen asetuksilla. Erillisiä verkkoalueita luodaan reitittimen asetuksiin määrittämällä virtuaalisia lähiverkkoja (VLAN eli Virtual Local Area Network). Virtuaaliset lähiverkot on määritelty standardissa IEEE 802.1Q[5]. Virtuaalisilla lähiverkoilla voidaan rajata helpommin liikennettä yhden verkkoalueen sisällä, jolloin myös palomuurisääntöjen tekeminen helpottuu. (Cisco, 2014) (Valtiohallinnon tietoturvallisuuden johtoryhmä, 2010)

Virtuaaliset lähiverkot ovat yhden sisäverkon aliverkkoja. Nämä aliverkot ovat omia loogisia kokonaisuuksia. Samaan aliverkkoon kuuluvat laitteet voivat sijaita fyysisesti eripuolilla verkkoa, jolloin laitteiden sijoittelusta ei tarvitse huolehtia. Aliverkot toimivat kuten tavallinen lähiverkko, joten niiden sisällä liikennöinti on helppoa, mutta muut aliverkot eivät voi keskustella keskenään ilman reititintä. Virtuaaliset verkot jakavat sisäverkon omiin verkkoalueisiin, jotka voivat keskustella keskenään, mutta

[5] http://www.ieee802.org/1/pages/802.1Q.html

näiden verkkojen väliset yhteydet ohjataan yleensä palomuurin kautta. Tällöin voidaan varmistaa verkkojen välisen liikennöinnin turvallisuutta. (Cisco, 2014)



Kuva 5: Virtuaalisten verkkojen hahmotelma verkkolaitteiden kanssa.

Kuvassa 5 on esimerkki, kuinka sisäverkko 192.168.0.0/25 on jaettu kahteen aliverkkoon 192.168.0.0/26 ja 192.168.0.64/26. VLAN 100 kuvastaa tätä ensimmäistä verkkoa ja VLAN 200 jälkimmäistä verkkoa. Nämä verkot ovat siis loogisesti toisistaan erillään ja kaikki liikenne verkkojen välillä kiertää reitittimen kautta. Reititystaulujen avulla voidaan ohjata liikenne haluttuun paikkaan ja mahdollisesti kierrättää palomuurin kautta. Jotta tämä tekniikka olisi mahdollista toteuttaa verkossa pitää myös kytkimien tukea tätä standardia. (Cisco, 2014)

3.1.2 Palomuuri

Palomuuri on laite ja/tai ohjelmisto, jonka tarkoituksena on rajoittaa ja suodattaa verkkoalueiden välistä liikennöintiä. Palomuurit voidaan jakaa karkeasti kahteen eri tyyppiin, tilallisiin ja tilattomiin. Näitä molempia palomuurityyppejä voidaan hyödyntää samaan aikaan. Molemmilla tyypeillä on omat vahvuutensa ja heikkoutensa, jolloin niiden yhdistäminen verkkoa rakennettaessa on hyödyksi. Samalla saadaan luotua kerrospuolustusta, jolloin yhden palomuurin murtaminen ei vielä anna pääsyä verkkoon. (Valtiohallinnon tietoturvallisuuden johtoryhmä, 2003)

Perusperiaate palomuurin sääntöjen luomiseen on kieltää ensin kaikki liikenne. Tämän jälkeen voidaan sallia vain haluttu liikenne. Näin saadaan helposti helpommin hallittavat palomuurisäännöt. Palomuurien suunnittelussa pitää ottaa huomioon myös niiden kahdennus. Kahdennuksella vältytään verkkoliikenteen katkeamiselta laitteen rikkoutuessa. (Valtiohallinnon tietoturvallisuuden johtoryhmä, 2003)

Tilattomalla palomuurilla tarkoitetaan pakettisuodatusta. Pakettisuodatus on nopeaa, koska siinä tutkitaan vain verkkoliikenteen pakettien otsikkokenttiä. Pakettisuodatukset säännöt ovat hyvin yksinkertaiset, jolloin niitä on helppo ylläpitää. Sääntöjen yksinkertaisuus tarkoittaa hyvää toimintavarmuutta, mutta tuo myös rajoitteita liikenteen rajoittamiseen. Pakettisuodatuksella ei voi esimerkiksi rajoittaa yksittäisen sovelluksen toimintaa. Yksinkertaisuudesta johtuen tilaton palomuuri on nopea, joten sitä käytetään verkkoliikenteen karkeaan suodatukseen. Tilaton palomuuri tarvitsee kuitenkin tilallisen palomuurin rinnalle, jotta ne voisivat yhdessä saavuttaa tarvittavan tietoturvallisuuden tason. (Valtiohallinnon tietoturvallisuuden johtoryhmä, 2003)

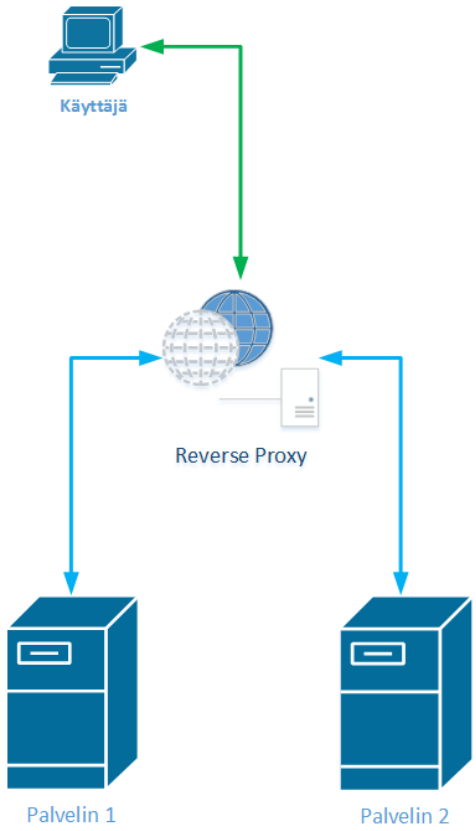
Tilallinen palomuuri toimii yleensä verkkoprotokollapinon yläkerroksissa. Palomuurin tilallisuudella tarkoitetaan, että se pitää yllä yhteyksien tiloja ja osaa dynaamisesti hallita omia sääntöjään. Dynaamisuus sallii palomuurin avata yhteyksiä sisäverkosta otettujen yhteyksien paluuviesteille. Palomuuri osaa huomata, mistä suunnasta yhteys on avattu ja tämän perusteella sallia tai estää liikenne. Tilallisten palomuurien kahdennuksessa pitää ottaa huomioon avoinna olevien yhteyksien tilojen reaaliaikainen kahdennus, jotta palomuurin rikkoutuessa avoinna olevat yhteydet eivät katkeaisi. (Valtiohallinnon tietoturvallisuuden johtoryhmä, 2003)

Näiden kahden erityyppisen palomuurin lisäksi on olemassa verkkosovelluksen paketteja tarkkaileva palomuuri. Tämän ei ole tarkoitus korvata edellä mainittuja palomuureja, vaan tuoda lisää pakettisuodatusta tarkemmalla tasolla. Perinteisemmät palomuurit tarkastelevat verkkoliikennettä yleensä kerroksilla kolme ja neljä. Verkkosovelluspalomuuri puolestaan tarkastelee liikennettä kerroksella seitsemän. Tällä tasolla voidaan liikenteestä päätellä, minkä sovelluksen liikennettä verkossa kulkee ja estää haluttujen sovelluksien toiminta verkossa. Haittapuolena näin tarkassa pakettien tarkastelussa on sen vaatima suuri prosessointitarve. Lisäksi sovelluksen tunnistaminen ei aina ole helppoa, varsinkaan jos mahdollinen hyökkääjä yrittää peittää jälkiänsä. (Woodberg, 2011)

3.1.3 Reverse proxy

Reverse proxyn tarkoituksena on tuoda lisää abstraktiota ja hallintaa verkkoliikenteeseen. Tämän tarkoituksena on toimia viestinvälittäjänä sovelluksen ja palvelimen välillä. Toiminta perustuu siihen, että sovellukselta tullut kutsu tulee ensin reverse proxylle, joka tekee tarvittavat kutsut oikealle palvelimelle. Reverse proxy odottaa palvelimelta vastaukset ja välittää nämä sovellukselle kuin ne olisivat tulleet

reverse proxylta. Näin saadaan peitettyä palvelimien näkyvyyttä ulkomaailmaan. Kuva 6 esittää reverse proxyn sijoittumisen verkkotasolla. (F5, 2016)



Kuva 6: Reverse proxyn sijoittuminen verkkotasolla.

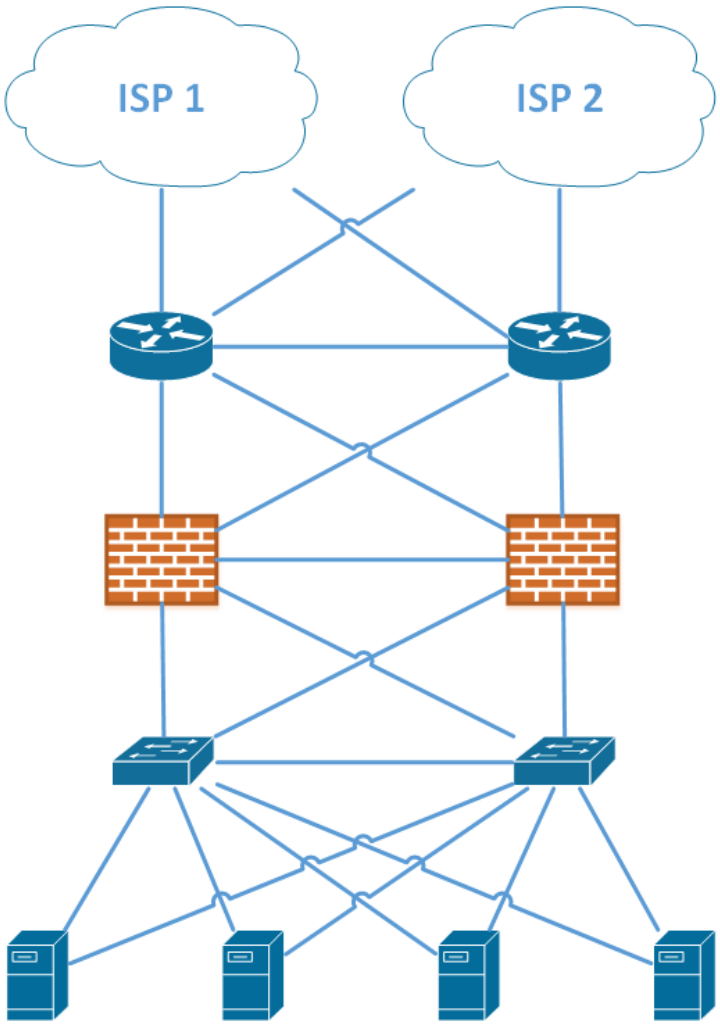
Kuvassa 6 käyttäjän ja reverse proxyn välinen liikenne on merkitty vihreällä. Käyttäjä näkee siis vain reverse proxyn osoitteen ja luulee, että tieto tulee tältä palvelimelta. Reverse proxy kuitenkin välittää nämä käyttäjältä tulleet kyselyt varsinaisille palvelimille ja välittää tiedot käyttäjälle, kuin ne olisivat tulleet reverse proxylta. Tämä reverse proxyn ja palvelinten välinen liikenne on kuvattu kuvaan sinisellä.

Reverse proxyä voidaan käyttää muun muassa kuormantasaajana tai ottamaan yhteyttä palvelimeen, joka on palomuurin takana. Tämä tuo siis lisää turvaa varsinaisen palvelimen turvaamiseen ja reverse proxyn läpi tulleita kyselyitä voidaan helpommin kontrolloida. Näin voidaan tehdä tiukempia palomuurisääntöjä oikeiden palvelinten suojaamiseen ja samalla vähennetään eri lähteiden määrää, jotka voivat liikennöidä palvelimille. Näiden lisäksi reverse proxyllä voidaan parantaa verkkosivujen suorituskykyä muun muassa ottamalla vastuu SSL:stä ja pakkaamalla sivuja. (F5, 2016)

3.2 Verkon varmistus

Verkon varmistamisella pyritään turvaamaan järjestelmän saatavuus. Verkon saatavuus saadaan turvattua, kun verkon laitteet ja yhteydet kahdennetaan. Kahdentamalla

palomuurit, verkon aktiivilaitteet sekä verkkoyhteydet saadaan verkosta rakennettua kestävä, jossa yksittäisen komponentin vikaantuminen ei haittaa. Kuten kaikessa tietoturvallisuudessa ja palvelun laadun varmistamisessa, SLA:n määrittäminen ratkaisee kahdentamisen laajuuden. Riskianalyysin perusteella tehdään päätökset, mihin riskeihin tulee varautua. (Valtiohallinnon tietoturvallisuuden johtoryhmä, 2010)



Kuva 7: Hahmotelma kahdennetusta verkosta.

Kuvassa 7 on hahmotelma kahdennetusta verkosta. Kuvasta huomaa kahdennuksen eri osa alueet eli laitteiden ja verkkoyhteyksien kahdennuksen. Esimerkin verkosta voi siis rikkoutua puolet laitteista, kunhan rikkoutuneet laitteet eivät ole samalla tasolla. Laitteen rikkoutuessa verkkoliikenne vain ohjataan toista reittiä, eikä vikaantumista pitäisi juurikaan huomata käyttäjän näkökulmasta. Näitä eri osa-alueiden kahdennuksia käydään läpi seuraavissa alikohdissa. Näiden lisäksi alikohdassa 3.2.3 käydään läpi kuormantasaajat, jotka osaavat tasata kuormaa eri palvelinten välillä. Tämä on siis enemmän kahdennetun palvelimen kuormittamiseen, mutta se kuitenkin tehdään verkkotasolla.

3.2.1 Laitteiden kahdennus

Verkon saatavuutta saadaan parannettua verkon aktiivilaitteiden kahdentamisella. Kahdentamalla laitteet saadaan verkosta poistettua pisteet, joiden vikaantuminen rikkoisi verkon toiminnan (Single Point of Failure). Kahdennetut laitteet mahdollistavat toisen laitteen rikkoutumisen ilman, että verkon toiminta häiriintyy. Riskianalyysin avulla saadaan selvitettyä, mikä osa verkosta pitää olla kahdennettuna ja minkälaisia seurauksia verkon katkeamisella on. (Valtiohallinnon tietoturvallisuuden johtoryhmä, 2010)

Verkkolaitteiden kahdennuksessa pitää myös huomioida reititystaulujen huolellinen suunnittelu sekä mahdollisten asetusten synkronointi laitteiden välillä. Kahdennusta voidaan nimittäin tehdä kahdella eri tavalla: joko niin, että molemmat laitteet ovat koko ajan päällä, tai siten, että varalaite on poissa käytöstä. Jälkimmäisessä tapauksessa varalaite on kuitenkin heti valmiina, jos se huomaa päälaitteen rikkoutumisen. Erityisen tärkeää laitteiden asetuksien synkronoinnissa on palomuurilla. Palomuurisääntöjen pitää olla aina ajan tasalla. Lisäksi tilallisen palomuurin tilatiedot tulee olla varalaitteen tiedossa tai muutoin pääpalomuurin vikaantuminen katkaisee sen hetkiset yhteydet. (Vartiainen, 2011)

Kahdennettujen laitteiden välillä vaihtuvaan viestinvaihtoon on olemassa useita protokollia ja menetelmiä riippuen verkkokerroksesta. OSI-mallin ensimmäisessä kerroksessa kahdennus tapahtuu useamman kaapelin avulla. Siirtokerroksessa käytetään muun muassa STP- (Spanning Tree Protocol) tai RSTP- (Rapid Spanning Tree Protocol) protokollia. Verkkokerroksessa pitää päivittää reititystaulut oikeiksi muuttuneen verkon takia. Reititystaulujen päivitys on kuitenkin normaali toimenpide reitittimille, jotka huomaavat muutoksia verkossa. Siirtokerroksessa vikaantumista ei välttämättä edes huomata, jos käytetään UDP-protokollaa. TCP-protokollan tapauksessa tilatiedot pitää synkronoida palomuurien ja kuormantasaajien kanssa, jos TCP-yhteyden ei haluta katkeavan. Siirtokerroksesta ylöspäin on yhteyden vikasietoisuuden vastuu protokollan tekijällä tai sovelluksella. Näihin kerroksiin verkkolaitteet eivät nimittäin ota kantaa. (Vartiainen, 2011)

Jotta verkkolaitteet saadaan pidettyä päällä sähkökatkojen aikana, pitää laitteille järjestää keskeytymätön virransyöttö (UPS, Uninterruptible Power Supply). Tämä voidaan hoitaa pienemmissä virtakatkoissa akustolla, mutta pidempiaikainen virransyöttö vaatii jo generaattorin. Virtakatkoihin pitää myös varautua laitteiden päässä, jotta ne osaavat palautua normaalitilaan uudelleen käynnistyessään. Verkkolaitteiden rikkoutumisen syy on monesti virtalähde. Paremmissa malleissa virtalähde onkin kahdennettu ja vikaantuessa se on mahdollista vaihtaa uuteen Hot swap -tekniikalla. (Valtiohallinnon tietoturvallisuuden johtoryhmä, 2010)

3.2.2 Verkkoyhteyksien kahdennus

Verkkoyhteyksien kahdennusta voidaan tehdä niin sisäverkossa, kuin myös Internet-yhteyden runkoverkkoyhteydessä. Kaapelointia tehdessä on siis otettava huomioon mahdolliset kahdennustarpeet. Kaapeloinnissa tehdyt virheet aiheuttavat myös saatavuusongelmia ja ne voivat rikkoitua ajan myötä. Kaapeloinnin kahdentamisessa tulisi huomioida, että kaapelit vedetään eri reittiä. Kaapelin tyyppi on myös merkitsevässä osassa. Normaalisti kaapeloinnissa käytetään kuparikaapeleita, jotka riittävät sisäverkon tarpeisiin. Nopeissa yhteyksissä ja etenkin runkoyhteyksissä voidaan käyttää valokuitua. Valokuidulla voidaan myös suojata laitteita ylijännitteen aiheuttamilta vahingoilta. (Valtiohallinnon tietoturvallisuuden johtoryhmä, 2010)

Sisäverkon verkkoyhteyksien kahdentamisella tarkoitetaan sisäverkon aktiivilaitteiden ja palomuurien välisten yhteyksien kahdennusta. Jotta verkkoyhteydet toimivat yhden laitteen rikkoutumisen jälkeen, verkkoliikenne pitää pystyä ohjaamaan myös toista reittiä. Tämä huomataan kuvassa 7, kun tarkastellaan aktiivilaitteiden yhteyksiä toisiinsa. Yhteyksiä tulee useita, mutta näin saadaan varmistettua verkkoliikenteen liikkuminen, vaikka jokin laite hajoaisi. Verkon kahdentamisessa otetaan samoin huomioon myös päätelaitteet. Palvelinten yhteydet on myös syytä kahdentaa, mutta esimerkiksi työntekijöiden omat tietokoneet eivät tätä yleensä tarvitse. Työntekijöiden omien tietokoneiden verkkoyhteyksien kahdentaminen tulisi helposti kalliiksi ja he voivat yleensä vain siirtyä toiseen paikkaan, jossa verkkoyhteys toimii. (Cisco, 2008)

Organisaatiot tarvitsevat usein pääsyn ulkomaailmaan tai ne tarjoavat palveluita sinne. Sisäverkon kahdennuksen lisäksi tarvitaan siis runkoverkkoyhteyden kahdennus, jotta Internet-palveluntarjoajan eli operaattorin (ISP, Internet Service Provider) verkkoyhteyden katkeaminen ei haittaisi palvelun saatavuutta. Kuvassa 7 on runkoyhteyden kahdennus toteutettu kahden eri operaattorin avulla (ISP 1 & ISP 2). Runkoyhteys voidaan siis toteuttaa, joko kahden eri operaattorin yhteydellä tai se voidaan toteuttaa yhden operaattorin palveluna. Kaikkialla kahden eri operaattorin yhteys ei kuitenkaan ole vaihtoehtona, koska esimerkiksi Suomessa operaattoreilla ei ole kaikkialla omaa verkkoinfraa. Tämän takia on alueita, joissa on pääasiassa vain yhden operaattorin toimintaa.

Kahden eri operaattorin vaihtoehto voi tulla kalliimmaksi, mutta se on varmempi, kuin käyttää vain yhtä operaattoria. Tässä vaihtoehdossa toisen operaattorin verkon vikaantuminen ei haittaa toista operaattoria ja palvelut pysyvät saatavilla. Pyydettäessä toiselta operaattorilta kahdennettua yhteyttä, pitää kuitenkin varmistua, että yhteys on todella kahdennettu. Operaattorit nimittäin vuokraavat yhteyksiä toisille operaattoreille ja kahdennuksessa pitää huomioida, että kahdennettu yhteys ei ole pääyhteyden tarjoavan operaattorin kaapeli. Kahdennuksessa tulisi myös varmistaa, että yhteydet tulevat eri kautta rakennukseen. Kaapelien eri reittien tarkoitus on estää niin sanotut kaivinkonevahingot, jossa kaivinkoneella katkaistaan kaapeli vahingossa.

Yhden operaattorin yhteyden kahdentamisessa pitää ottaa huomioon, että varayhteys ei tule viereisessä kaapelissa. Tämä vaihtoehto on kuitenkin heikompi vaihtoehto, koska pää- ja varayhteys kulkevat saman operaattorin runkoverkossa. Tällöin operaattorin runkoverkon vikaantuessa ovat molemmat yhteydet poikki.

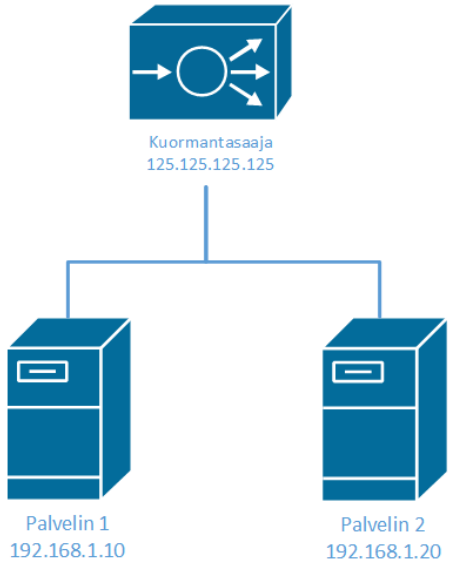
Runkoverkkoyhteyksien kahdennus hoidetaan yleensä kaapeliyhteyksillä. Varsinkin isojen konesalien yhteydet ovat usein kuidulla, koska niihin tarvitaan nopeita yhteyksiä. Kuitu mahdollistaa myös hyvin pitkien välimatkojen kaapelit, koska kuitutoistinten etäisyydet voivat olla erittäin pitkiä (kymmeniä kilometrejä). Tällaisten yhteyksien rakentaminen on kuitenkin kallista, joten pienemmät toimistoverkot on kahdennettu usein langattomalla varayhteydellä. Esimerkiksi 3G- ja 4G-yhteyksien nopeudet ovat jo riittävällä tasolla, jotta niitä voitaisiin käyttää varayhteyksinä pienemmissä toimistoissa.

3.2.3 Kuormantasaajat

Kuormantasaajan tehtävänä on tasata kuormaa palvelinten välillä. Kun palvelin halutaan kahdentaa, voidaan palvelimelle tulevat kyselyt ohjata kuormantasaajan kautta. Kuormantasaaja tarkkailee siihen liitettyjen palvelinten kuormitusta sekä tarkastelee vastaavatko palvelut normaalisti. Jos huomataan, että jokin palvelu ei vastaa tai toinen palveluista on kovan kuormituksen alla, ohjataan kyselyt toiselle palvelimelle. Tämän tekniikan avulla käyttäjät eivät edes huomaa kummalle palvelimelle kyselyt tulevat. Kuormantasaaja mahdollistaa myös useamman kuin kahden palvelimen kyselyiden välittämisen. Näin voidaan siis helposti lisätä palvelun skaalautuvuutta käyttäjämäärien kasvun myötä. (Linux Virtual Server, 1998)

Toteutettu maksujärjestelmä on tilaton. Tilattomuus helpottaa kuormantasaajan toimintaa, koska kyselyitä ei tarvitse ohjata aina samalle palvelimelle. Tilallisuus tuo omat haasteensa kuormantasaajan toimintaan. Tilallisissa kyselyissä pitää pystyä ohjaamaan kyselyt oikealle palvelimelle, jossa on aiemmin avattu istunto avonaisena. Tämä tarkoittaa, että kuormantasaajan pitää myös pitää yllä kyselyjen tiloja. (Linux Virtual Server, 1998)

Kuvassa 8 nähdään kuinka kahdennetulle palvelulle ohjatut kyselyt tulevat kuormantasaajan kautta. Palvelimille voidaan antaa sisäverkon osoitteet ja vain kuormantasaaja vastaa palvelun yhteiseen IP-osoitteeseen. Kuvasta huomataan, että nyt kuormantasaaja on kriittisessä pisteessä, koska sen kautta ohjataan kaikki kyselyt palvelimille. Kuormantasaajia suunniteltaessa pitää myös kahdentaa kuormantasaajat. (Linux Virtual Server, 1998)



Kuva 8: Kuormantasaaja kahdelle palvelimelle.

3.3 Organisaatioiden sisäverkkojen yhdistäminen

Tutkittavassa maksujärjestelmässä on esimerkiksi varmennuspalvelu, jota ei haluta sijoittaa DMZ-alueelle, vaan se pitää laittaa turvallisemmalle verkkoalueelle. Varmennus halutaan suojata mahdollisimman hyvin, ja kaikkia yhteyksiä tähän palveluun valvotaan tarkasti. Jotta varmennuspalveluun saataisiin otettua yhteys ulkoverkosta, on yksi mahdollisuus pystyttää VPN-tunneli (Virtual Private Network). Toinen tapa olisi käyttää MPLS-tekniikkaa (Multiprotocol Label Switching), joka ei kuitenkaan salaa liikennettä. (Valtiohallinnon tietoturvallisuuden johtoryhmä, 2010)

MPLS-tekniikalla saadaan yhdistettyä organisaatioiden sisäverkkoja yhteen. Jos liikenne halutaan vielä salata tällä välillä, pitää käyttää lisäksi jotain muuta protokollaa salaamaan liikenne tällä välillä. MPLS-tekniikka on operaattorien tarjoama tekniikka, jonka käyttäminen saattaa tulla kalliiksi. Tästä syystä VPN-tunnelien käyttö on paljon yleisempää, koska se on mahdollista toteuttaa organisaation IT:n avulla. VPN soveltuu myös moneen eri käyttötarkoitukseen ja sen vahva salausta säilyttää verkkoyhteyden luotettavuuden ja eheyden. (Valtiohallinnon tietoturvallisuuden johtoryhmä, 2010)

VPN-tunnelin avulla saadaan turvattua yhteys varmennuspalvelun ja varmentajan välillä sekä valvottua, että vain sallittu kohde voi avata yhteyden. Tämä on hyvin yleinen tapa yhdistää kahden eri organisaation verkkoja. VPN-tunnelin sisällä voidaan liikennöidä millä tahansa protokollalla. Ulospäin se näkyy aina salattuna. VPN-yhteyksiä on mahdollista rakentaa eri tavoin riippuen minkälaista suojausta ollaan hakemassa. VPN-tunnelin voi tehdä esimerkiksi kahden palvelimen välille suoraan tai organisaatioiden reunareitittimien välille. Jälkimmäisessä tapauksessa liikenne ei kulje enää VPN-tunnelin sisällä organisaatioiden sisäverkossa. Jälkimmäinen vaihtoehto on hyvä myös

siitä syystä, että organisaatioiden palomuurit voivat tarkastella VPN-yhteyden yli tulevaa liikennettä. (Valtiohallinnon tietoturvallisuuden johtoryhmä, 2003)

VPN mahdollistaa useita erilaisia tunnelointimenetelmiä. Nykyisin käytetyimpiä tunnelointiprotokollia ovat PPTP (Point-to-Point Tunneling Protocol), L2TP (Layer Two Tunneling Protocol) ja IPSec (Internet Protocol Security). Tunnelointi-protokollat toimivat joko OSI-mallin toisella tai kolmannella kerroksella. Edellä mainituista protokollista PPTP ja L2TP toimivat siirtokerroksessa ja IPSec verkkokerroksessa. Nämä tunnelointi-protokollat mahdollistavat IP-verkkoliikenteen toiminnan paketoimalla IP-paketin oman IP-paketin sisään. Tämä ylimääräinen IP-paketointi poistetaan tunnelin toisessa päässä, jossa paketti jatkaa kohteeseen ilman VPN-tunnelin tarjoamaa salausta. (Microsoft, 2001)

4. TIETOKANTA

Tietokanta on toteutetun maksujärjestelmän tärkein osa, johon kaikki tieto tallennetaan. Tietokannan tietojen eheys ja saatavuus ovat siis erittäin tärkeitä järjestelmän toiminnan kannalta. Eheyden menetys tarkoittaa, että tietoihin ei voi enää luottaa. Tietokantapalvelimen kaatuminen tai verkkohäiriöt aiheuttavat saatavuuden menettämisen. Jos muilla palveluilla ei ole yhteyttä tietokantaan, ei tietoja pystytä näyttämään käyttöliittymällä tai kyselyihin ei pystytä antamaan vastauksia. Esimerkiksi varmennuspalvelu ei voi antaa vastauksia, jos se ei saa yhteyttä kantaan.

Tässä luvussa käsitellään tietokantaa yleisesti, mutta tarkemmassa käsittelyssä keskitytään Microsoftin SQL[6]-tietokantaan. SQL-kieli on standardisoitu, mutta jokainen SQL-tietokanta eri valmistajalta pitää sisällään myös niille ominaisia eroavaisuuksia. Microsoftin SQL-tietokanta valittiin tarkempaan tarkasteluun, koska se oli käytössä projektin aikana.

Neljännessä luvussa käydään kolme tärkeintä kokonaisuutta tietokantojen tietoturvaa mietittäessä. Ensimmäisenä kohdassa 4.1 käydään läpi tietokannan kahdennus sekä varmuuskopioiden tekeminen. Tämän jälkeen siirrytään oikeuksien hallintaan kohdassa 4.2, jossa käydään läpi miten oikeuksia tulisi käsitellä ja kenelle niitä pitäisi myöntää. Lopuksi kohdassa 4.3 käydään läpi tietojen salaus tietokannassa.

4.1 Varmistaminen

Tietokannan varmistamisessa on kaksi pääasiaa, joista tulee huolehtia. Varmuuskopioiden avulla saadaan tietokanta palautettua haluttuun tilaan esimerkiksi virheen takia. Varmuuskopioita tulee säilyttää niin, että tietokantapalvelimen rikkoutuminen ei hävitä tietoja, vaan varmuuskopioista voidaan palauttaa tietokanta tilaan ennen rikkoutumista. Varmuuskopioilla saadaan huolehdittua tietokannan eheydestä virhetilanteissa. (Cherry, 2012)

Varmuuskopioiden lisäksi tulee huomioida tietokannan kahdentaminen. Kahdentamisella saadaan turvattua tietokannan saatavuutta tilanteissa, joissa yksi tietokannoista on alhaalla virheen tai rikkoutumisen johdosta. Kahdennuksessa pitää kuitenkin huomioida tietokannan tietojen synkronointi.

[6] http://www.microsoft.com/fi-fi/server-cloud/products/sql-server/

4.1.1 Varmuuskopiointi

Varmuuskopioiden tarkoituksena on palauttaa tietokannan tiedot takaisin, mikäli tiedot esimerkiksi korruptoituvat tai tietokantapalvelin hajoaa. Palvelimia pystyttäessä laitetaan kovalevyt usein RAID-tilaan. RAID:lla pyritään yleensä varmistamaan, että yhden tai useamman kovalevyn rikkoutuminen ei vielä hävitä tietoa. Tämä riippuu tietenkin RAID-tilasta. Esimerkiksi RAID 1-tila tarkoittaa peilausta, jossa kaksi kovalevyä pitävät tietonsa täysin samanlaisena. Tämä ei kuitenkaan tarkoita sitä, että varmuuskopioita ei pitäisi ottaa. RAID:n on tarkoitus pitää järjestelmä toiminnassa kovalevyn hajotessa, mutta se ei auta silloin kun koko palvelin tuhoutuu. (Cherry, 2012)

Varmuuskopioiden tiheyteen vaikuttaa järjestelmässä olevan tiedon kriittisyys. Maksujärjestelmässä käsitellään rahaa, joten siksi on jo hyvin tärkeää, että tietoa ei häviä järjestelmästä. Varmuuskopioita tulee siis ottaa mahdollisimman tiheästi. Varmuuskopioiden tekeminen vie kuitenkin hetken ja sen valmistumisen jälkeen on jo voinut syntyä paljonkin uutta tietoa. Täyden tietokantavarmuuskopion saa luotua esimerkiksi komennolla: (Cherry, 2012)

```
BACKUP DATABASE <tietokannan_nimi> TO DISK=<tiedostopolku>
```

Täysiä varmuuskopioita kannattaakin luoda harvemmin ja tämän lisäksi voidaan luoda differentiaalikopioita. Differentiaalikopio sisältää kaiken muuttuneen tiedon viimeisen täyden varmuuskopion jälkeen. Täyden varmuuskopion ja differentiaalikopioiden avulla voidaan siis palauttaa tietokanta haluttuun tilaan. Differentiaalivarmuuskopion saa otettua esimerkiksi komennolla: (Cherry, 2012)

```
BACKUP LOG <tietokannan_nimi> TO DISK=<tiedostopolku>
```

Varmuuskopioiden tekeminen pitää kuitenkin automatisoida, jotta ne tulee tehtyä tarpeeksi usein. Tähän liittyy myös vanhojen varmuuskopioiden poistaminen. Jotta levytila ei täyty, pitää huolehtia tarpeeksi vanhojen varmuuskopioiden poistamisesta. Varmuuskopioita tulisi myös säilyttää toisella palvelimella tai mahdollisimman kaukana oikeasta palvelimesta. Tällöin varmuuskopiot säilyvät isommankin onnettomuuden sattuessa. (Cherry, 2012)

Varmuuskopioinnin automatisointiin Microsoftin SQL-palvelimella löytyy useita eri vaihtoehtoja, kuten *Maintenance Plan*:in eli ylläpitosuunnitelman tekeminen. Tämän avulla voidaan määrittää miten usein tehdään varmuuskopioita ja minkälaisia. Varmuuskopion on mahdollista kirjoittaa levylle tai lähettää suoraan verkon yli toiselle palvelimelle. Ylläpitosuunnitelma antaa myös mahdollisuuden salata varmuuskopiot. Salaamalla varmuuskopiot huolehditaan, etteivät asiattomat henkilöt saa tietoja käsiinsä, vaikka he pääsisivät varmuuskopioihin käsiksi. Varmuuskopiot jäävät usein huomioimatta tai niitä saatetaan kuljettaa paikasta toiseen, jolloin ne on mahdollista kaapata. (Cherry, 2012)

4.1.2 Kahdennus

Tietokannan kahdennuksen voi monella eri tavalla. Toteutetussa projektissa oli käytössä Microsoftin SQL – palvelimen ominaisuus AlwaysOn, jonka avulla tietokantapalvelin saadaan klusteroitua ja hajautettua useampaan palvelimeen. (Microsoft, 2016)

Klusteroinnin tarkoituksena on hajauttaa tietokantapalvelimet useammalle palvelinalustalle. Yhden palvelimen vikaantuessa klusterointi osaa ottaa käyttöön toisen palvelimen. Tällöin tietokanta saadaan hajautettua ja kahdennettua, jolloin saavutetaan korkea saatavuus tietokannalle. Klusterointi vaatii tietokantojen tietojen synkronointia eri palvelimien välillä jotta kaikilla palvelimilla olisi aina samat tiedot. (Microsoft, 2016)

Tietojen synkronointiin käytetään synkronista tietojen kahdennusta. Synkronisuus takaa sen, että palvelimilla on varmasti samat tiedot. Vasta kun tiedot ovat molemmilla palvelimilla, voidaan tieto kuitata tallennetuksi. Tämä aiheuttaa hieman hitautta tietojen tallennukseen, mutta sen avulla varmistetaan tietojen pysyminen ajan tasalla kaikilla palvelimilla. (Microsoft, 2016)

Tietokannan käyttäjän ei tarvitse tietää klusteroinnista mitään, vaan se ottaa aina yhteyttä määriteltyyn domain nimeen. Domainiin on linkitetty kaikkien tietokantapalvelimien IP-osoite, jolloin aina sen hetken päätietokantapalvelin vastaa kyselyihin. (Microsoft, 2016)

4.2 Oikeudet

Oikeuksien hallinta ja pääsynvalvonta ovat tärkeimpiä huomioitavia asioita. Resurssien käyttöä tulee rajoittaa vain oikeutetuille henkilöille. Oikeuksia pitää myös osata myöntää oikeisiin kohteisiin pääsynhallinnan menetelmillä. Menetelmiä on muutamia erilaisia, jotka sopivat hieman eri tilanteisiin. Näiden lisäksi myös käyttäjien tunnistaminen pitää toteuttaa luotettavasti. (OWASP, 2009)

4.2.1 Oikeuksien jakaminen

Oikeuksien jakamisessa tulee muistaa aina vähimpien oikeuksien periaate (Principle of least privilege). Vähimpien oikeuksien periaatteessa henkilöille annetaan vähimmät oikeudet, jotka he tarvitsevat toimiensa suorittamiseen. Tämä tulee myös huomioida itse ohjelmissa, joita ajetaan palvelimilla. Ohjelmilla ei myöskään tulisi olla liian isoja oikeuksia, jotta mahdollinen hyökkääjä ei pystyisi hyödyntämään ohjelman oikeuksia hyökkäyskoodia ajettaessa. (OWASP, 2009)

Ohjelmien ja käyttäjien lisäksi myös tietokannan oikeuksien hallinta tulee tehdä huolella. Tietokannan käyttäjille myönnetään usein liian isot oikeudet, jolloin

tietokannan eheys saattaa vaarantua. Ohjelmille pitäisi yleensä riittää vain taulujen luku ja kirjoitusoikeus, jolloin ohjelmien tunnuksilla ei voisi tehdä isompia tietokannan muokkauksia. Luku- ja kirjoitusoikeudet on mahdollista eriyttää eri prosesseille, jolloin yhden prosessin vaarantuminen ei anna kuin toisen näistä oikeuksista. (OWASP, 2009)

Microsoftin SQL -palvelin tarjoaa kaksi tapaa käyttää tunnuksia tietokannassa. Toinen tavoista on käyttää Microsoftin AD-tunnuksia (Active Directory) (Microsoft, 2016) ja toinen on luoda tietokantaan omia tunnuksia. Näistä vaihtoehdoista Microsoftin AD-tunnuksien käyttö auttaa tunnuksien hallitsemista huomattavasti, kun käyttäjille ei tarvitse luoda erikseen tunnuksia tietokantaan. AD:n avulla käyttäjät voivat käyttää tietokantaa samoilla tunnuksilla, kuin he kirjautuvat esimerkiksi omille koneilleen. (OWASP, 2009)

4.2.2 Identiteetin todentaminen

Identiteetin todentamisella eli autentikoinnilla tarkoitetaan prosessia, jossa tarkastetaan käyttäjän identiteetti. Identiteetin todentamisessa käytetään erilaisia menetelmiä, jotka voidaan jaotella kolmeen pääryhmään. Nämä pääryhmät ovat ”jotain, mitä henkilö tietää tai muistaa”, ”jotain, mitä henkilöllä on hallussa” ja ”jotain, mitä henkilö on”. (Valtiohallinnon tietoturvallisuuden johtoryhmä, 2006)

Jotain, mitä henkilö tietää tai muistaa, tarkoittaa esimerkiksi salasanaa tai PIN-koodia. Tämä on nykyään yleisin tapa tunnistautua palveluihin. Salasanan ja tunnuksen käyttäminen kirjautumiseen on helppo ottaa käyttöön, mutta se ei ole kuitenkaan turvallisin tapa. Salasana on helppo vakoilla toiselta ja se saatetaan laittaa liian helpoksi, jolloin se on helppo muistaa, mutta myös helppo arvata tai murtaa. (Valtiohallinnon tietoturvallisuuden johtoryhmä, 2006)

Jotain, mitä henkilöllä on hallussaan, on esimerkiksi pankkikortti, puhelin tai salasanoja generoiva laite. Tämä on huomattavasti turvallisempi tapa tunnistaa käyttäjä, kuin salasana. Nyt henkilöllä pitää olla fyysisesti jokin laite käytössään eikä niitä ole niin helppo saada käsiinsä. Niiden käytöstä tulee yleensä kustannuksia, koska ne pitää jotenkin valmistaa ja toimittaa henkilöille. Lisäksi laitteen kuljetuksen aikana turvallisuus voi vaarantua, jos joku saa sen tuolloin käsiinsä. Nykyään yleistyy mobiiliapplikaatiot, jotka toteuttavat salasanoja generoivan laitteen ominaisuuden. Tällöin henkilölle ei tarvitse toimittaa erillistä laitetta, vaan hänen pitää vain ladata sovellus sekä mahdollisesti toimittaa sovelluksen tunniste. (Valtiohallinnon tietoturvallisuuden johtoryhmä, 2006)

Jotain, mitä henkilö on, tarkoittaa biometristä tunnistusta. Biometrinen tunnistus voi perustua erilaisiin ominaisuuksiin, joita henkilöllä on. Yleisimpiä ominaisuuksia ovat sormenjälki, kädenjälki tai -muoto sekä silmän iiris. Tämän hyvänä puolena on se, että biometrinen tunnistus kulkee aina henkilön mukana. Biometristä ominaisuutta on myös

melkein mahdotonta varastaa ainakaan ilman huomaamista. (Valtiohallinnon tietoturvallisuuden johtoryhmä, 2006)

Kun henkilön identiteetin todentaminen pohjautuu vain yhteen edellä mainituista ryhmistä, pidetään tunnistamista heikkona. Vahva tunnistaminen vaatii vähintään kahden todentamistavan hyödyntämisen. Esimerkiksi kortilla maksamisessa, käyttäjä esittää oman sirukorttinsa ja syöttää PIN-koodin. Nykyään myös puhelimiin on tullut mahdollisuus hyödyntää biometristä tunnistusta. Kun tämän biometrisen tunnistuksen yhdistää esimerkiksi salasanan syöttöön saadaan tehtyä vahva tunnistautuminen. (Valtiohallinnon tietoturvallisuuden johtoryhmä, 2006)

4.2.3 Pääsynvalvonta

Pääsynvalvontaa voidaan toteuttaa erilaisin perustein. Periaatteet voidaan jakaa kolmeen eri ryhmään, jotka ovat yksilöpohjainen pääsynvalvonta (Discretionary access control, DAC), pakotettu pääsynvalvonta (Mandatory access control, MAC), roolipohjainen pääsynvalvonta (Role-based access control, RBAC). (OWASP, 2015)

Pääsynvalvontamatriisi (Access control matrix) on yleisin pääsynvalvontamalli yksilöpohjaisessa pääsynvalvonnassa. Matriisi koostuu riveistä, jotka kuvaavat järjestelmän eri käyttäjiä tai ryhmiä. Sarakkeissa on kuvattu jokin resurssi, jolloin tämän sarakkeen alapuolella oleviin soluihin tulee tieto kunkin käyttäjän oikeuksista kyseiseen resurssiin. Pääsynvalvontamatriisi on kuvattu taulukossa 1. Yhden sarakkeen tietoja kutsutaan myös pääsynvalvontalistaksi, joka siis kuvaa vain yhden resurssin oikeudet kaikille käyttäjille. (Hu, et al., 2006)

Taulukko 2: Pääsynvalvontamatriisi.

	Resurssi 1	**Resurssi 2**	**Resurssi 3**
Käyttäjä 1	luku, kirjoitus	luku	luku
Käyttäjä 2	luku, kirjoitus	luku	-
Käyttäjä 3	luku	luku, kirjoitus	luku, kirjoitus
Ryhmä 1	luku, kirjoitus	luku, kirjoitus	-

Pakotetussa pääsynvalvonnassa ei ole erillisiä pääsynvalvontalistoja. Pääsynvalvonta perustuu erilaisiin tasoihin, jotka on ennalta määritelty. Tasot voivat olla esimerkiksi erittäin salainen, salainen, luottamuksellinen ja rajoitettu. Käyttäjät eivät itse huolehdi tiedostojen käyttöoikeuksista, vaan tämä on keskitetyn käyttövalvonnan esimerkiksi käyttöjärjestelmän tehtävä. Käyttäjälle on annettu tietty taso, johon hänellä on oikeus ja sen mukaan myös hänen käyttämien resurssien oikeus määräytyy. Taulukkoon 3 on kuvattu henkilön, jonka taso on salainen, oikeudet eri tason resursseihin. (Hu, et al., 2006)

Taulukko 3: Sääntöpohjaisen pääsynvalvonnan oikeudet resursseihin, kun turvaluokitus on salainen.

	Erittäin salainen	**Salainen**	**Luottamuksellinen**	**Rajoitettu**
Oikeudet	kirjoitus	luku, kirjoitus	luku	luku

Taulukosta 3 huomataan, kuinka henkilön oikeudet rajoittuvat vain omaan tasoon ja siitä alaspäin lukuoikeuden muodossa. Kirjoitusoikeus hänellä on puolestaan omasta tasostaan ylöspäin. Tätä kutsutaan Bell-La Padula -malliksi. Mallissa on siis kaksi sääntöä *no read-up* sekä *no write down*. *No read-up* tarkoittaa, että käyttäjä ei voi lukea hänen tasoaan korkeammalla olevia resursseja. *No write-down* tarkoittaa, että käyttäjä ei voi kirjoittaa omaa tasoaan matalamman luokituksen tietoa. (Hu, et al., 2006)

Roolipohjaisessa pääsynvalvonnassa on käyttäjälle annettu rooleja, joita hän tarvitsee tehtäviensä suorittamiseen. Rooleille on puolestaan määritelty käyttövaltuudet resursseihin. Käyttäjälle voidaan määritellä useita rooleja, jolloin hänen käyttövaltuutensa resursseihin kasvaa. Rooleja voidaan helposti lisätä ja poistaa käyttäjältä hänen toimenkuvansa muuttuessa, jolloin vähimpien oikeuksien periaatetta on helppo noudattaa. Taulukossa 4 on kuvattu esimerkki muutaman roolin tapauksesta. (Hu, et al., 2006)

Taulukko 4: Roolipohjaisen pääsynvalvonnan esimerkki yliopistossa.

	Kurssi ilmoittautuminen	**Harjoitustyöt**	**Tentti**
Opiskelija	luku, kirjoitus	kirjoitus	kirjoitus
Assistentti	luku	luku	-
Opettaja	luku	luku	luku

Taulukosta 4 nähdään, miten eri rooleilla saattaa olla samoja oikeuksia resursseihin. Tässä on käytetty esimerkkinä yliopiston erilaisia rooleja. Esimerkissä on kuitenkin kuvattu vain muutama erilainen rooli, jota tällaisessa ympäristössä voisi olla. Esimerkiksi ylemmän tason roolilla voidaan tehdä samat asiat, kuin alemman tason roolilla. Tämän lisäksi on resursseja, joihin on oikeudet vain osalla rooleista.

4.3 Salaus

Tietokannan salauksella saadaan turvattua tietojen luottamuksellisuutta jos joku on luvattomasti saanut tietokannan haltuunsa. Salauksessa pitää kuitenkin muistaa, että se tarvitsee aina osan prosessorin kuormasta, jolloin suorituskyky saattaa hieman hidastua. Tämän takia kannattaakin miettiä, mitkä tiedot tarvitsevat salauksen ja mitkä eivät.

Tämän lisäksi voidaan valita riittääkö tietojen salaus tiivistyksellä (Hash) vai tarvitaanko salausalgoritmeja (Encryption). Samoin pitää miettiä, miten vahvaa salausta tarvitaan, jotta osataan valita sopiva algoritmi. Mitä vahvempaa salausta käytetään, sitä enemmän prosessoriaikaa tarvitaan salauksen purkuun. (Cherry, 2012)

4.3.1 Tiivistäminen ja salaus

Tietojen tiivistäminen on kevyempi prosessi, mutta sillä voidaan vain luoda tiiviste. Tätä luotua tiivistettä ei voida purkaa takaisin selkokielelle. Tiedon tiivistämistä käytetään usein esimerkiksi käyttäjien salasanojen tallentamiseen. Salasanoja ei tarvitse koskaan saada takaisin selkokielelle, koska kirjautumisen yhteydessä annettu salasana tiivistetään uudestaan ja sitä verrataan tietokannassa olevaan salasanan tiivisteeseen. Jos tiivisteet ovat samat, niin oikea salasana on syötetty. Tiivistämisessä on otettava kuitenkin huomioon tarvittavan vahvan tiivistealgoritmin käyttö. Heikoimmat tiivistealgoritmit on pystytty jo murtamaan ja osaan on tehty valmiita tiivistesanakirjoja. Tiivistesanakirjoissa on siis valmiiksi tiivistetty eri merkkiyhdistelmiä, jolloin niitä voidaan verrata saatuun tiivisteeseen ja näin purkamaan tiivisteen muodostuksessa käytetty merkkijono. (Cherry, 2012)

Sanakirjahyökkäyksiä vastaan voidaan suojautua käyttämällä tiivistyksessä suolaa. Suolalla tarkoitetaan jonkin tunnetun salaisen tiedon lisäämistä salasanaan ennen salasanan tiivistämistä. Suolan tarkoituksena on pitää salasana suojattuna, vaikka joku saisikin sanakirjalla auki tiivistetyn salasanan. Tällöin hyökkääjä saa selville salasanan ja suolan yhdistetyn tiedon, josta ei voida päätellä oikeata salasanaa ilman, että tietää suolaa. Suola pitääkin suojata hyvin, jotta mahdollinen hyökkääjä ei saa sitä haltuunsa. Tämän lisäksi tiivistämisessä voi tapahtua niin sanottu törmääminen. Törmäämisellä tarkoitetaan sitä, että kaksi täysin eri merkkijonoa muodostaa saman tiivisteen. Vaikka tämä on harvinaista, niin se on kuitenkin mahdollista tapahtua. Tällä hetkellä SHA 2 256bit- ja SHA 2 512bit -tiivisteitä pidetään turvallisina. (Cherry, 2012)

Salausalgoritmin käyttö tiedon salaamiseen on parempi valinta, jos tietoa halutaan käyttää. Tässä tapauksessa käytetään yleensä symmetristä salausta. Symmetrisessä salauksessa käytetään samaa avainta sekä tiedon salaukseen että purkamiseen. Tällöin salausavaimen pitäminen turvassa on tärkeää. AES on nykyään yleinen symmetrinen salausalgoritmi, joka on luokiteltu vahvaksi salaukseksi. (Cherry, 2012)

4.3.2 Tiedon salaus

Microsoft SQL-palvelin tarjoaa versiosta 2008 ylöspäin muutaman valmiin funktion, joiden avulla voidaan tietokannan taulussa oleva tieto salata. Nämä funktiot ovat `EncryptByCert`, `EncryptByKey`, `EncryptByPassPhrase` ja `EncryptByAsymKey`. (Cherry, 2012)

EncryptByCert ottaa parametrikseen sertifikaatin tunnisteen sekä merkkijonon tai muuttujan, joka halutaan salata. Funktio salaa annetun tiedon sertifikaatin julkisella avaimella. Vastaavasti tiedon saa purettua käyttämällä funktiota DeCryptByCert, joka purkaa salatun tiedon sertifikaatin yksityisellä avaimella. Tässä käytetään siis epäsymmetristä salausta. Esimerkki tämän funktion käytöstä on ohjelmassa 1. (Microsoft, 2016)

```
--Salataan merkkijono ”Salattava tieto” muuttujaan @Salattu
DECLARE @Salattu VARBINARY(128)
   = ENCRYPTBYCERT(CERT_ID(’Sertifikaatti’), ‘Salattava tieto’)
```

Ohjelma 1: MS SQL salaus sertifikaatin julkisella avaimella. (Cherry, 2012)

EncryptByKey käyttää symmetristä salausta. Funktiolle annetaan ensin käytettävän avaimen GUID ja tämän jälkeen salattava tieto. Salatun tiedon saa purettua DecryptByKey -funktiolla. Esimerkki tämän käytöstä on ohjelmassa 2. (Microsoft, 2016)

```
--Avataan symmetrinen avain
OPEN SYMMETRIC KEY Avain
--Salataan merkkijono ”Salattava tieto” muuttujaan @Salattu
DECLARE @Salattu VARBINARY(128)
   = ENCRYPTBYKEY(Key_GUID(’Avain’), ‘Salattava tieto’)
```

Ohjelma 2: MS SQL salaus symmetrisellä avaimella. (Cherry, 2012)

EncryptByPassPhrase salaa tiedon kolminkertaisella DES-algoritmillä käyttämällä 128 bittistä avainta. Funktiolle voi siis antaa maksimissaan 128 bittiä pitkän salausavaimen tai -lauseen, josta muodostetaan symmetrinen salausavain. Edellisten tapaan tiedon saa purettua käyttämällä funktiota DecryptByPhassPhrase. Funktion käyttö on esitetty ohjelmassa 3. (Microsoft, 2016)

```
--Salauslause, jota käytetään salaamiseen
DECLARE @SalausLause NVARCHAR(128) = ’Salauslause tiedon salaamiseen’
--Salataan merkkijono ”Salattava tieto” muuttujaan @Salattu
DECLARE @Salattu VARBINARY(256)
   = ENCRYPTBYPASSPHRASE(@SalattuLause, ’Salattava tieto’)
```

Ohjelma 3: MS SQL salaus salauslauseella 3DES algoritmilla. (Cherry, 2012)

EncryptByAsymKey salaa tiedon käyttämällä asymmetrista avainta. Muuten funktio toimii samalla tavalla kuin funktio EncryptByKey. Salauksen purkaminen tapahtuu vastakkaisella funktiolla DecryptByAsymKey. Salausfunktiolle annetaan ensin asymmetrinen avain ja tämän jälkeen salattava tieto. Funktion kutsu on esitetty ohjelmassa 4. (Microsoft, 2016)

```
--Salataan merkkijono ”Salattava tieto” muuttujaan @Salattu
DECLARE @Salattu VARBINARY(256)
```

```
= ENCRYPTBYAsymKey(AsymKey_ID('AsymmetrinenAvain'), 'Salattava tieto')
```

Ohjelma 4: MS SQL salaus käyttämällä asymmetrista avainta. (Cherry, 2012)

Kaikille edellä mainituille tavoille salata tieto on yhteistä niiden rajoitukset salattavan tiedon kokoon. Kaikki funktiot palauttavat *VARBINARY*-tyyppisen paluuarvon, jonka maksimi suuruus on 8000 tavua. Tämä siis rajoittaa kuinka suuri salattava tieto voi olla.

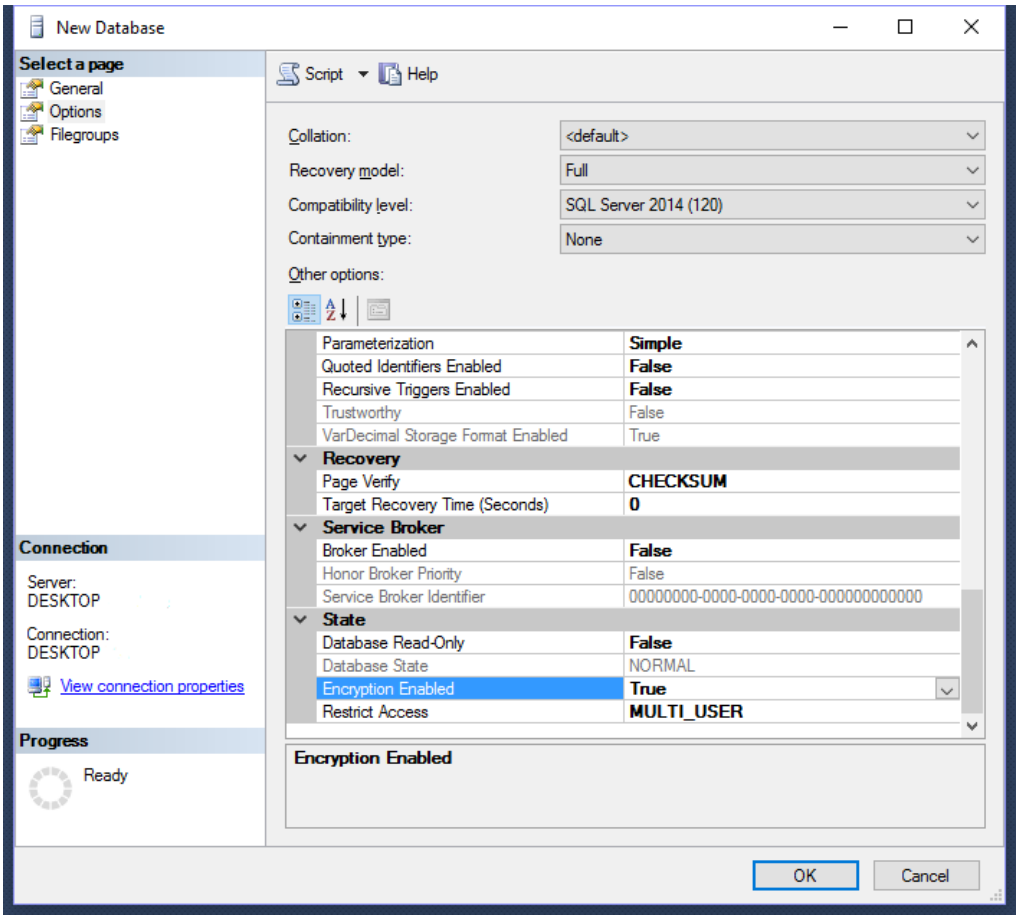
4.3.3 Tietokannan salaus

Edellä kuvatut tiedon salauksen tavat salaavat tiedon tietokannan taulun sisällä. Tällöin tietokannan rakenne, taulujen nimet ja metatiedot jäävät salaamatta. Myös nämä tiedot on mahdollista salata salaamalla koko tietokanta. Tähän on Microsoftin SQL-palvelimen versiosta 2008 lähtien ollut suora tuki, jonka saa helposti käyttöön. Tätä salaustekniikkaa kutsutaan TDE:ksi (Transparent Data Encryption). (Cherry, 2012)

TDE-tekniikan hyvä puoli on se, että sillä saa salattua koko tietokannan ja se on helppo ottaa käyttöön. Tietokanta pidetään aina salattuna levyllä, jolloin myös siitä otetut varmuuskopiot ovat salattuna. Tiedot puretaan salauksesta vasta, kun ne luetaan palvelimen muistiin. Huono puoli salauksessa on tietysti sen tuoma kuorma palvelimen suorittimelle. TDE-salaus aiheuttaa lisäkuormaa kaikille samalla instanssilla ajettaville tietokannoille, koska se salaa myös *tempdb*-tietokannan. *Tempdb*-tietokanta on väliaikaiseen tallennukseen käytetty tietokanta, johon voidaan tallentaa tietoa jonkin kyselyn sisällä. Jos tiedot ovat kuitenkin kriittisiä ja ne vaativat salausta, tulee tämä salauksen aiheuttama lisäkuorma huomioida palvelinalustaa valittaessa. Kuvasta 9 nähdään, kuinka TDE-salaus saadaan asetettua päälle tietokannalle käyttämällä Microsoftin SQL Server Management Studio -työkalua. (Cherry, 2012)

Kuvassa 9 näkyvän ikkunan saa auki, kun luo uutta tietokantaa tai katsoo tietokannan asetuksia (Properties). Salausasetuksen saa päälle, kun menee vasemmassa reunassa näkyvälle asetukset (Options) välilehdelle. Salaus asetetaan päälle vaihtamalla ”Encryption Enabled” arvoon True. Tämä asetus näkyy kuvassa valittuna sinisellä taustavärillä.

Salaus on mahdollista saada päälle myös käyttämällä T-SQL-komentoja. Tämä on esitetty tarkemmin ohjelmassa 5. Ensimmäisessä vaiheessa luodaan master-tietokantaan salausavain ja sertifikaatti. Tämän jälkeen haluttu tietokanta salataan valitulla salausalgoritmilla, ja käyttämällä aiemmin luotua sertifikaattia. Lopuksi viedään muutokset tietokantaan ja käynnistetään itse salaus. (Cherry, 2012)



Kuva 9: Tietokannan salauksen ottaminen käyttöön Microsoftin SQL Server Management Studiolla.

```
--Luodaan master tietokantaan salausavain ja sertifikaatti
USE master
CREATE MASTER KEY ENCRYPTION BY PASSWORD = ‘Salausavain’
CREATE CERTIFICATE SalausSertifikaatti WITH SUBJECT = ‘Salaussertifikaatti’
GO
--Vaihdetaan käyttöön tietokanta, joka halutaan salata
USE SalattuTietokanta
--Luodaan tietokannan salaus halutulla salausalgoritmilla ja aiemmin
--luodulla sertifikaatilla
CREATE DATABASE ENCRYPTION KEY
WITH ALGORITHM = AES_128
ENCRYPTION BY SERVER CERTIFICATE SalausSertifikaatti
GO
ALTER DATABASE SalattuTietokanta
SET ENCRYPTION ON
GO
```

Ohjelma 5: Salauksen ottaminen käyttöön Microsoftin SQL-palvelimella käyttäen T-SQL-kieltä. (Cherry, 2012)

Tietokannan salauksen lisäksi on huolehdittava salausavaimen turvallisesta säilyttämisestä. Se ei saa päätyä muille, kuin tarpeellisille henkilöille, ja se tulee säilyttää varmassa tallessa. Jos salausavain hukkuu, salattuja tietoja ei saada enää auki. Tästä syystä salausavaimista ja sertifikaateista tulee olla varmuuskopiot. Varmuuskopioita tulisi olla vähintään kaksi, joista toinen on jossain kauempana eri rakennuksessa ja/tai eri kaupungissa. (Cherry, 2012)

4.3.4 Proseduurien ja funktioiden salaus

Tietojen salauksen lisäksi Microsoft SQL-tietokanta mahdollistaa myös ohjelmakoodin salauksen. Tämä tarkoittaa, että SQL-proseduurit ja funktiot voidaan tuoda kantaan salattuina, jolloin niitä ei voi muokata tietokannasta käsin. Tämä antaa hieman suojaa koodin logiikan paljastumiseen. Tämä salaus saadaan tehtyä käyttämällä ”WITH ENCRYPTION” lausetta. Proseduurin salaaminen on esitetty ohjelmassa 6.

```
CREATE PROCEDURE SalattuProseduuri
@muuttuja AS INT
WITH ENCRYPTION
AS
--Tästä alkaa proseduurin koodi
```

Ohjelma 6: MS SQL prodesuurin salaus. (Cherry, 2012)

Edellä esitetty tapa salata SQL-proseduuri tai -funktio ei ole kuitenkaan kovin vahva. Salausmetodi, jota Microsoft käyttää salaamaan proseduurit ja funktiot, on jo murrettu. Tämä tarkoittaa myös sitä, että tarjolla on kolmannen osapuolen ohjelmia, joiden avulla voidaan poistaa näiden proseduurien ja funktioiden salaus.

5. PALVELINOHJELMISTO

Palvelinohjelmiston tarkoituksena on tarjota rajapinnat tietojen käsittelyyn. Tämän lisäksi palvelinohjelmistot sisältävät paljon logiikkaa, jonka avulla eri toiminnallisuuksia on tehty. Tietoturvamielessä palvelinohjelmisto toimii käyttäjän ja tietokannan välissä. Siten tavalla saadaan kasvatettua abstraktiotasoa tietokannasta sekä luotua suojakerros tietokannan eteen. Tietokantaan tulisikin antaa pääsy vain näiden palvelinohjelmistojen kautta, jotta tietokantaan ottavien määrää saisi karsittua ja yhteyksiä olisi helpompi hallita.

Tämä luku koostuu kolmesta kohdasta, jotka käsittelevät palvelinohjelmiston suunnittelua sekä ylläpitoa tietoturvallisuuden näkökulmasta. Kohdassa 5.1 käydään läpi tietoturvallisen ohjelmoinnin asioita. Näihin kuuluu olennaisena osana OWASP Top Ten -listaus verkkosovellusten yleisimmistä tietoturvariskeistä. Ohjelmoinnin jälkeen kohdassa 5.2 käsitellään rajapintoja, oikeuksien hallintaa sekä syötteiden validointia. Lopuksi kohdassa 5.3 käydään läpi palveluiden ylläpitoon ja hallintaan liittyviä asioita. Näihin kuuluu muun muassa palveluiden kahdentaminen sekä tilan ja lokien hallinta.

5.1 Ohjelmoinnin hyvät käytännöt

Palvelinohjelmiston turvallisuus on hyvin pitkälti kiinni siitä, miten hyvin ohjelmoijat noudattavat hyviä käytäntöjä. Näiden lisäksi ohjelmoijilla pitäisi olla vähintään perustietämys yleisimmistä haavoittuvuuksista. OWASP:n (Open Web Application Security Project) verkkosivut tarjoavat kattavasti tietoa erilaisista haavoittuvuuksista niin yleisesti kuin myös ohjelmointikielikohtaisesti. OWASP-organisaation tarkoituksena on parantaa ohjelmistojen tietoturvaa tarjoamalla tietoa ilmaiseksi kaikille. (OWASP, 2016)

5.1.1 OWASP Top Ten

OWASP Top Ten on listaus kymmenestä kriittisimmästä verkkosovellusten tietoturvariskeistä. Vaikka listaus on kohdistettu verkkosovelluksille, ovat monet riskit oleellisia myös muissa ohjelmissa. Kirjoitushetkellä viimeisin listaus on tehty vuonna 2013. Listausta päivitetään aina muutaman vuoden välein, vaikka osa haavoittuvuuksista on ollut listalla pitkäänkin. Yksi näistä listalla pitkään olleista riskeistä on SQL-injektio. Seuraavaksi käsitellään tämän listan tärkeimmät riskit, jotka

vaikuttavat tutkittavan maksujärjestelmän tietoturvallisuuteen. Kokonaisuudessaan listaus on taulukossa 5. (OWASP, 2013)

Taulukko 5: OWASP Top Ten lista. (OWASP, 2013)

A1	Injection
A2	Broken Authentication and Session Management
A3	Cross-Site Scripting (XSS)
A4	Insecure Direct Object Reference
A5	Security Misconfiguration
A6	Sensitive Data Exposure
A7	Missing Function Level Access Control
A8	Cross-Site Request Forgery (CSRF)
A9	Using Components with Known Vulnerabilities
A10	Unvalidated Redirects and Forwards

Uusimmalla listalla ensimmäisellä paikalla on injektio. Injektiolla tarkoitetaan, että käyttäjältä tulevaa parametria ei tarkisteta ollenkaan, jolloin se voi sisältää esimerkiksi ajettavaa ohjelmakoodia. Hyökkääjän on mahdollista kirjautua järjestelmään sisään tai saada luottamuksellista tietoa järjestelmästä injektioiden avulla. Tästä syystä käyttäjältä tulevat syötteet tulee aina tarkistaa ja huomioida, ettei näitä laiteta suoraan esimerkiksi SQL-komennon sisään. Syötteiden tarkistuksista käydään läpi myöhemmin tässä luvussa. (OWASP, 2013) (Cherry, 2012) (Singhal, et al., 2007)

Toisena listalla on rikkinäinen autentikointi ja session hallinta. Tällä tarkoitetaan sitä, että hyökkääjän on mahdollista saada haltuunsa käyttäjän autentikointitiedot esimerkiksi URL-osoitteesta. Tästä syystä URL-parametreihin ei tulisi koskaan laittaa mitään arkaluonteista tietoa, koska linkkiä jakaessa nämä tiedot näkyvät kaikille. Myös session aikakatkaisu ja salasanojen tallentaminen liittyvät tähän riskiin. Salasanojen tallentaminen käsiteltiin jo aikaisemmassa luvussa ja autentikointia ja session hallintaa käydään läpi myöhemmin tässä luvussa. (OWASP, 2013)

Suorat objektien viitteet ovat hyvin kriittisiä tietoturvan kannalta ja ne löytyvät listalta neljänneltä paikalta. Viitteet tietokantaobjekteihin tulisi olla sellaisessa muodossa, että niitä ei voida arvata, vaikka niitä näkyisikin käyttäjälle. Tästä syystä siihen kiinnitettiin erityistä huomiota myös tutkittavassa maksujärjestelmässä. Viitteinä käytettiin satunnaisesti generoituja viitteitä, jolloin viitteestä ei ole mahdollista päätellä esimerkiksi objektien määrää järjestelmässä, ja niiden arvaaminen on myös hyvin vaikeaa. (OWASP, 2013)

Viidentenä kohtana on tietoturva-asetusten väärinkonfigurointi. Tämä voi tuntua perusasialta, mutta jos sitä ei tehdä, on hyökkääjällä mahdollisuus tunkeutua järjestelmään. Tähän kuuluu sekä asetusten tarkastaminen että oletuskäyttäjien ja -

salasanojen vaihtaminen. Kuudennessa kohdassa mainitaan kriittisen tiedon altistus hyökkääjälle. Tämä tarkoittaa, että salausta on käytettävä kaiken kriittisen tiedon salaamiseen tallennuspaikalla, että kuljetettaessa tietoa. Esimerkiksi kaikkien rajapintojen tulisi olla SSL-salattuja, jotta hyökkääjä ei voi kuunnella liikennettä ja poimia sieltä kriittistä tietoa. Maksujärjestelmässä korttien numerot, jos niitä käytetään, tulisi suojata hyvin, jotta hyökkääjä ei voisi tehdä ostoksia muiden korteilla. (OWASP, 2013) (Singhal, et al., 2007)

Listan seitsemäs kohta tarkoittaa käytännössä rajapintakutsujen oikeuksien tarkastamista. Rajapintakutsuissa pitäisi aina tarkistaa oikeudet, jotta kirjautumaton käyttäjä ei saa sen kautta tietoja. Tämän lisäksi pitää muistaa tarkistaa, että käyttäjä saa nähdä vain hänelle kuuluvan tiedon. Tätä asiaa käsitellään tarkemmin kohdassa 5.2. (OWASP, 2013) (Singhal, et al., 2007)

Yhdeksäs kohta listalla tarkoittaa ohjelman käyttämien komponenttien, kirjastojen tai muun tällaisen osan tietoturva-aukon hyödyntämistä. Tästä syystä ohjelman käyttämät komponentit tulisi pitää ajan tasalla ja seurata niitä koskevia uutisia, jotta mahdolliset tietoturva-aukot havaitaan ja voidaan tehdä väliaikaisia korjauksia, kunnes komponentista tulee päivitetty versio. Esimerkiksi OpenSSL kirjaston vakava haavoittuvuus Heartbleed[7] oli paljonkin uutisissa sen vakavuuden ja kirjaston suuren suosion takia. (OWASP, 2013)

Listan loput haavoittuvuudet ovat lähinnä www-ohjelmistoihin liittyviä tietoturvariskejä. Listan kolmas, eli Cross-Site Scripting (XSS), toimii hieman aiemmin mainittujen injektioiden tapaan. Tämä ilmenee silloin, kun käyttäjän syötteitä ei tarkasteta ja käyttäjä saa syötettyä esimerkiksi JavaScript-koodia syötteeksi. Syötteiden validointia käsitellään tarkemmin kohdassa 5.2. (OWASP, 2013) (Singhal, et al., 2007)

Kahdeksantena listalla on Cross-Site Request Forgery (CSRF), joka tarkoittaa kyselyiden väärentämistä. CSRF hyökkäys on mahdollinen jos sivustolla on rajapinta, jolle voi antaa suoraan esimerkiksi URL-parametreina tarvittavat tiedot jonkin operaation suorittamiseen. Hyökkääjä piilottaa tämän kutsun omalle sivustolleen ja odottaa, että käyttäjä, joka on kirjautunut hyökkäyskohteen sivustolle, tulee hyökkääjän sivustolle. Käyttäjä suorittaa tällöin hyökkääjän kutsun käyttäjän omilla kirjautumistiedoilla. Tämän saa estettyä käyttämällä sessiokohtaista tokenia jokaisessa kutsussa. Tokenia ei tulisi myöskään antaa suoraan URL-parametrina, vaan viedä esimerkiksi http-kyselyn sisällä. (OWASP, 2013)

Viimeisenä listalla on vielä uudelleenohjauksien validointi. Tällä tarkoitetaan, että kaikki sivuston uudelleenohjaukset tulevat validoida, jotta hyökkääjä ei voi antaa parametrina omaa URL-osoitetta, jonne käyttäjä siirtyy. Tällöin käyttäjä voisi epähuomiossa luulla hyökkääjän tekemää sivustoa oikeaksi ja antaa omia tietojaan

[7] http://heartbleed.com/

hyökkääjälle. Hyvällä sivustosuunnittelulla tällaisia virheitä ei pitäisi päästä syntymään. Joskus uudelleenohjauksia kuitenkin tarvitaan, ja silloin pitää muistaa huolehtia, että käyttäjä ei voi antaa tätä osoitetta missään vaiheessa. Syötteiden validointi tulee myös tässä tapauksessa esille, jos uudelleenohjauksen parametri on pakko antaa esimerkiksi URL-parametrina. (OWASP, 2013)

5.1.2 Turvallinen ohjelmointi

Turvallisen ohjelmoinnin periaatteet pitäisi olla projektin ohjelmoijilla sekä suunnittelijoilla tiedossa, jotta vältytään aiemmassa kohdassa käsitellyiltä hyökkäyksiltä. Kaikki tietoturvallisuus perustuu tietoturvan kolmeen osa-alueeseen: luottamuksellisuuteen, saatavuuteen ja eheyteen. OWASP:in turvallisen ohjelmoinnin periaatteiden listaan kuuluvat seuraavat osa-alueet: hyökkäys-alueen minimointi, turvalliset vakioarvot, oikeuksien hallinta, kerrospuolustus, turvallinen kaatuminen, muihin palveluihin luottaminen, tehtävien jakaminen, turvallisuuden luominen peittämällä, yksinkertaisuus ja tietoturvan korjaaminen. (OWASP, 2016)

Hyökkäysalueen minimoinnilla tarkoitetaan, että rajapintoihin tulee antaa oikeudet vain rajatulle käyttäjäryhmälle. Kaikki rajapinnan laajennokset tarjoavat uuden mahdollisen hyökkäyskohteen. Erityisesti jos käyttäjä voi antaa syötteitä, pitää nämä tarkastaa erityisellä huolellisuudella. Syötteiden validointia käydään tarkemmin läpi alakohdassa 5.2.2. Muihin palveluihin luottaminen voidaan rinnastaa syötteiden tarkastukseen. Monet järjestelmät välittävät tietoa toisilleen, mutta vaikka kyse on kolmannen osapuolen järjestelmästä, ei tämän toimintaan tulisi luottaa täysin. Muissa järjestelmissä voi aina ilmetä erilaisia vikoja, jolloin sieltä saadut syötteet voivat olla vahingollisia. (OWASP, 2016)

Turvallisilla vakioarvoilla halutaan varmistaa, että palvelu toimii oletusarvoisesti turvallisesti. Näihin voi kuulua esimerkiksi salasanan tai session vanhenemisaika. Oikeuksien hallinta on jo käsitelty aiemmassa kohdassa 4.2. Tärkeintä on kuitenkin muistaa vähimpien oikeuksien periaate, jossa käyttäjille annetaan vähimmät oikeudet jotka vaaditaan tehtävien suorittamiseen. (OWASP, 2016)

Tietoturvallisuuden suunnittelussa tulisi myös huomioida kerrospuolustuksen periaatteet. Puolustuksessa ei tulisi luottaa vain yhteen kerrokseen, joka tekee kaiken tietoturvan. Puolustusta tulee hajauttaa useampaan eri kohtaan, jotta yhden puolustuksen murruttua ei olisi vielä täyttä pääsyä järjestelmään. Puolustukseen liittyy myös ohjelman turvallinen kaatuminen poikkeustilanteissa. Ohjelman pitäisi aina kaatua turvallisesti ja mahdollisesti kirjoittaa lokeihin mitä on tapahtunut. Näin saadaan tietoon ohjelman poikkeuksellinen toiminta, jolloin tätä varten voidaan tehdä oma poikkeuskäsittely. Tähän liittyy myös OWASP:in tietoturvan korjaaminen, jolla tarkoitetaan juurikin tietoturvaan liittyvien asioiden korjaamista. Korjaaminen tulisi aina tehdä huolella, jotta korjaus ei aiheuttaisi uutta tietoturvariskiä järjestelmälle. (OWASP, 2016)

Tehtävien jakamisella saadaan tietoturvallisuudesta huolehtimista jaettua useammalle henkilölle. Jakaminen tulisi tehdä niin, että kukaan ei yksin vastaa kokonaisesta prosessista. Prosessia pitää hoitaa vähintään kaksi henkilöä, jolloin kukaan ei voi yksin tehdä mitään, mitä joku toinen ei huomaisi. Tällöin esimerkiksi salausavaimien säilyttäminen tulisi olla paikassa, jonka avaamiseen tarvitaan kaksi henkilöä. (OWASP, 2016)

Tietoturvallisuuden yksinkertaisuus ja asioiden peitteleminen liittyvät toisiinsa. Tietoturvan ei pitäisi ikinä perustua asioiden peittelemiseen, koska nämä salaisuudet voivat aina paljastua, jolloin koko tietoturva on vaarassa. Esimerkiksi salausavaimen kovakoodaaminen ja sen peittäminen ohjelmakoodiin on iso riski tietoturvallisuudelle. Ohjelmakoodia voi nimittäin kohtuullisen helposti kääntää ajettavasta binääritiedostosta. Tätä kutsutaan takaisinmallinnukseksi (Reverse Engineering). Tällöin hyökkääjän on mahdollista selvittää salausavain ja murtaa järjestelmän tekemä salaus. Yksinkertaisuutta tulisi myös suosia erityisesti tietoturvallisuuden kanssa. Suunnittelun yksinkertaistamisessa käytetäänkin KISS-periaatetta (Techopedia, 2016), joka on tullut tutuksi jo 1900-luvun puolivälistä. Liian monimutkaiset järjestelmät ovat hankalia ylläpitää ja niiden toimintaa voi olla hankala seurata, jolloin tietoturva-aukkojen riski kasvaa. (OWASP, 2016)

5.1.3 Koodikatselmointi

Koodikatselmoinnilla tarkoitetaan kirjoitetun ohjelmakoodin lukemista ja tarkastamista toisen henkilön toimesta. Joku muu kuin koodin kirjoittaja siis katsoo muutokset läpi ja antaa mahdollisia korjausehdotuksia. Katselmoinnilla pyritään estämään ohjelmavirheiden päätyminen lopulliseen ohjelmaan sekä kontrolloimaan minkälaisesta ohjelmakoodista lopullinen ohjelma koostuu. Koodikatselmoinnin tärkeys korostuu erityisesti tietoturvakriittisissä ohjelmissa, joissa ohjelmavirheet voivat aiheuttaa vakavia seurauksia. (Huston, 2016)

Katselmointia pidetään raskaana prosessina, jonka takia sitä ei haluta kuin kriittisissä projekteissa. Ennen katselmoinnin tavat olivatkin hieman raskaat, mutta nykyään ne voidaan tehdä hyvinkin kevyesti. Raskaat koodikatselmoinnit ovat tarpeellisia, kun ohjelmassa ei ole varaa minkäänlaisiin virheisiin esimerkiksi lentokoneissa tai muissa laitteissa, jotka vikaantuessa voisivat aiheuttaa vaaraa ihmiselle. Nykyään on kehitetty monia erilaisia tapoja suorittaa katselmointia. Näistä tavoista voi aina valita omaan tarkoitukseen sopivan katselmointitavan, jolloin työmäärä vastaa myös tarkoitustaan. Katselmointiin on myös olemassa työkaluja, jotka auttavat katselmoijaa tai tekee jopa omia havaintoja koodista. Toteutetun järjestelmän versionhallintaan käytettiin Git-versionhallintaa[8] (Git, 2016). Git-ohjelmassa on koodikatselmointiin sopiva ominaisuus

[8] https://git-scm.com/

PR (Pull Request) (Git, 2016), jonka avulla ohjelmoija voi viedä omasta haarastaan uutta koodia esimerkiksi versionhallinnan päähaaraan. (Huston, 2016)

Git:in PR-toiminnon avulla voidaan tarkastella uusi koodi ennen kuin se yhdistetään muun koodin kanssa. PR-toiminto on hyvin tuettu useissa Git-versionhallinnan verkkosivuilla, jolloin uuden tarkastaminen voidaan tehdä selaimella. PR:ää tarkasteltaessa verkkosivulla on merkitty muuttuneet tiedostot. Tiedostoja tarkastelemalla nähdään aina vanha ja muuttunut versio tiedostosta. Työkalut osaavat myös näyttää kaikki muuttuneet kohdat, jolloin ihmisen ei tarvitse etsiä näytä kohtia käsin. PR-sivulla muut ohjelmoijat voivat jättää kommentteja ja parannusehdotuksia koodiin, jolloin PR:n tehnyt ohjelmoija voi korjata kohdat. Jos koodi on hyväksyttävää ja mahdolliset korjaukset on korjattu, voidaan PR:lle antaa hyväksyntä ja yhdistää muun koodin kanssa. Avoimenlähdekoodin Git-versionhallinta GitHub[9] on hyvä esimerkki PR -toiminnan toteuttamisesta. (Git, 2016)

5.2 Rajapinnat

Rajapinnat ovat ohjelmiston tärkeimpiä osia. Niiden kautta on mahdollista syöttää tietoa ohjelmistolle ja suorittaa erilaisia operaatioita. Oikeuksien hallinnointi on tärkeää, jotta vain sallitut henkilöt voivat käyttää rajapinnan toimintoja tai saavat vain heille tarkoitetun tiedon. Oikeuksia myönnettäessä pitää muistaa minimiperiaate eli annetaan pienimmät oikeudet, jolla henkilö voi tehdä tarvittavat toimet. Oikeuksissa pitää myös huomioida niiden sulkeminen, jos henkilö ei niitä enää tarvitse tai on siirtynyt muihin tehtäviin. Oikeuksia käydään läpi tarkemmin alakohdassa 5.2.1. (OWASP, 2015)

Oikeuksien lisäksi käyttäjältä tulevat syötteet tulee validoida huolella. Validoinnilla tarkoitetaan syötteen tarkastusta, jotta se pitäisi sisällään vain sallittuja merkkejä. Esimerkiksi SQL-injektiot ovat hyvin yleisiä. SQL-injektio on mahdollista sen takia, että käyttäjältä tulevaa syötettä ei tarkasteta. Näin käyttäjän on mahdollista syöttää omaa koodiaan ajettavaksi ja aiheuttaa pahoja vahinkoja. Syötteiden validointi käsitellään tarkemmin alakohdassa 5.2.2. (OWASP, 2016)

5.2.1 Oikeudet

Rajapintojen käyttöä tulee rajoittaa, jotta voidaan seurata mitä komentoja on ajettu sekä rajoittamaan rajapinnan käyttö vain sallituille henkilöille. Toteutetussa maksujärjestelmässä on monia rajapintoja. Esimerkiksi varmennuksella on oma osoitteensa ja porttinsa, jossa se keskustelee varmennusprotokollan mukaisin sanomin. Tämä rajapinta on kuitenkin helppo suojata, koska sinne saa ottaa yhteyttä vain yhdestä osoitteesta ja tämäkin yhteys on suojattu VPN-tunnelilla. Muut rajapinnat ovat

[9] https://github.com

puolestaan pääosin toteutettu REST-periaatteella. (Maksujärjestelmäprojektiryhmä, 2015)

REST-rajapintojen suojauksessa lähdetään liikkeelle SSL-salauksesta, jolla saadaan salattua liikenne rajapinnan käyttäjän ja rajapinnan välillä. Salatun yhteyden läpi voidaan antaa esimerkiksi käyttäjätunnus ja salasana jokaisen kutsun mukana. Hieman turvallisempi tapa on kuitenkin kuljettaa mukana autentikointitokenia, jonka saa onnistuneen kirjautumisen jälkeen. Tokenille voidaan antaa vanhenemisaika. Tokenin saaminen hyödyttää vain vanhenemisaikaan asti tai kunnes uusi tokeni on myönnetty. (OWASP, 2015)

Autentikointitiedot pitää kuljettaa turvallisesti käyttäjän ja rajapinnan välillä. Tämän takia niitä ei tulisi sijoittaa HTTP:n GET-metodin parametriksi, vaan mieluummin POST-metodin tai keksien sisällä. URL-parametrit voi nimittäin näkyä verkkopalvelimen lokeissa. Lokeja tarkastelemalla voidaan tällöin saada haltuun tarvittavat autentikointitiedot. (OWASP, 2015)

Jotta tarvittava autentikointitoken on mahdollista saada, pitää ensin kirjautua onnistuneesti sisään. Kirjautumissivu on tietomurroissa ensimmäinen kohde, josta yritetään etsiä heikkouksia. Ensiksi tulisi varmistua, ettei kirjautumista voida yrittää loputonta määrää. Tällä tavalla saadaan estettyä sanakirjahyökkäyksiä. Käyttäjälle pitää kuitenkin antaa mahdollisuus syöttää salasana muutamaan otteeseen, jos ensimmäisillä kerroilla sen on syöttänyt väärin. Eräs tapa on eksponentiaalisesti kasvattaa aikaa, joka täytyy odottaa, kunnes sama käyttäjä voi yrittää syöttää salasanaa uudestaan. Esimerkiksi kolmannen väärän salasanan jälkeen tunnus menee lukkoon viideksi minuutiksi. Tämän jälkeen jos syöttää salasanan väärin tunnus menee lukkoon kymmeneksi minuutiksi. Tätä voidaan aina kasvattaa ja mahdollisesti tehdä jonkinlainen hälytys järjestelmän ylläpitäjälle. (OWASP, 2015)

Kirjautumiseen tarvitaan myös palvelu, joka tarkistaa annetun käyttäjätunnuksen ja salasanan. Ensimmäisenä pitää kuitenkin päättää, missä tunnuksia halutaan säilyttää. Loppukäyttäjien tunnukset tallennetaan usein järjestelmän käyttämään tietokantaan, jossa salasanoista säilytetään vain niiden tiivisteet. Toinen yleinen tapa on käyttää jotain identiteetinhallintajärjestelmää. Tämä on helpointa ottaa käyttöön järjestelmän sisäisille käyttäjille, joilla on jo organisaation sisäiset tunnukset. Tällaisia identiteetinhallintajärjestelmiä ovat esimerkiksi Microsoftin AD (Active Directory) (Microsoft, 2016) tai LDAP (Lightweight Directory Access Protocol) (Howes, 1995).

5.2.2 Syötteiden validointi

Syötteiden validointi tulisi huomioida erityisesti ulkomaailmaan näkyvissä rajapinnoissa, mutta myös sisäisissä rajapinnoissa. Sisäisiä rajapintoja käyttävät yleensä muut yrityksen työntekijät, joiden ei ajatella tekevän tällaisia hyökkäyksiä. Jos

työntekijällä on esimerkiksi erimielisyyksiä työantajan kanssa tai hän haluaa muuten saada luvatonta tietoa käsiinsä, voi hän yrittää oman koodin ajamista järjestelmään. Sisäisiä rajapintoja on kuitenkin helpompi valvoa, koska sinne on pääsy vain sisäverkon käyttäjillä. Ulkomaailmaan näkyviä rajapintoja pitää puolestaan suojata erityisellä huolella, koska sieltä voi kuka tahansa yrittää anonyymisti käyttää rajapintoja omiin tarkoituksiinsa.

Validointia voidaan tehdä kahdella tapaa. Joko niin, että tarkistetaan, ettei syöte sisällä määrättyjä sanoja, tai niin, että se sisältää vain listan mukaisia sallittuja merkkejä. Ensimmäinen tapa on niin sanottu *black list* -validointi, jota käytetään usein. Tämä ei kuitenkaan ole tehokkain tapa validoida syötettä, koska kiertomahdollisuuksia on monia ja tätä listaa pitää muistaa ylläpitää. Jälkimmäistä tapaa kutsutaan *white list* -validoinniksi. *White list* -tapauksessa syötteen merkit pitää löytyä listalta, jolloin vain sallitut merkit läpäisevät validoinnin. Tämä on yleensä toteutettu säännöllisten lausekkeiden (Regular expression[10]) avulla. Säännölliset lausekkeet mahdollistavat tiukkojenkin validointien tekemisen, jos odotettu syöte on rakenteeltaan aina säännönmukainen. Esimerkiksi sähköpostiosoite, postinumero tai henkilötunnus ovat hyvin säännönmukaisia rakenteita ja hyväksyvät vain ennalta määrätyt merkit. Merkkien validoinnin lisäksi syötteen pituus tulisi olla rajattu, jotta liian pitkiä syötteitä ei voisi edes antaa. (OWASP, 2016)

5.3 Palvelut

Palveluilla tarkoitetaan juurikin itse palvelinohjelmistoa. Tutkittavana oleva järjestelmä koostuu useista palveluista, jotka toimivat keskenään ja tarjoavat eri ominaisuuksia järjestelmässä. Esimerkiksi molemmat käyttöliittymät ovat omia palveluitaan, koska ne tarjoavat omat ominaisuutensa käyttäjille. Varmennuspalvelu on myös yksi palveluista. (Maksujärjestelmäprojektiryhmä, 2015)

5.3.1 Kahdennus

Järjestelmä on haluttu hajauttaa useisiin erillisiin palveluihin, jotta eri palvelut saataisiin omiin verkkoalueisiin ja mahdollisesti jokaisen voi laittaa ajoon omaan palvelinalustaan. Kuorman kasvaessa on myös mahdollista kasvattaa järjestelmän palvelinkapasiteettia helposti. Kapasiteettia saadaan kasvatettua laittamalla palvelinohjelmistosta uusi instanssi ajoon uudelle palvelimelle. Skaalautuvuuden avulla ei jouduta maksamaan turhasta ja kapasiteettia voidaan lisätä vasta tarpeen mukaan. (Maksujärjestelmäprojektiryhmä, 2015)

Istuntojen hallitseminen tuo omat haasteensa palvelinohjelmistossa. Istunnon siirtäminen käyttäjän laitteeseen helpottaa rajapintojen suunnittelua, sekä ohjelmiston

[10] http://regexr.com/

skaalautuvuutta. REST-rajapinnat tulisikin aina suunnitella tilattomiksi, jotta niiden käyttäminen olisi mahdollisimman helppoa ja rajapinnan palvelinkapasiteettia saadaan skaalattua helposti käyttäjämäärän kasvaessa. (Doglio, 2015)

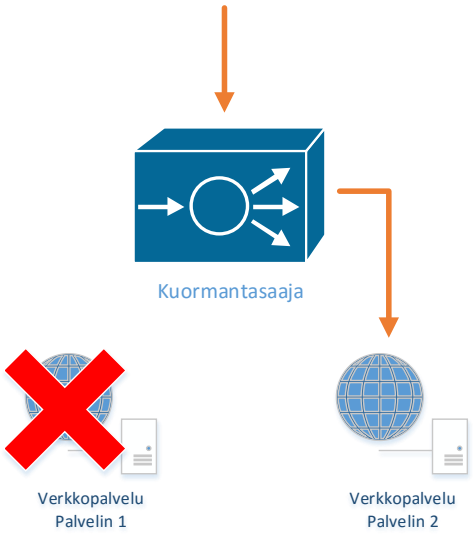
Tutkittavana olevassa järjestelmässä kaikki tieto on tallennettuna tietokantaan. Tietokannan tietoja ei laiteta välimuistiin, vaan tieto haetaan aina uudestaan kannasta, jotta saadaan aina ajantasaisinta tietoa. Tietokannan kahdennus täytyykin tehdä hieman eri tavalla, koska kaikilla tietokannan instansseilla pitää olla aina sama tieto. Tietojen synkronoinnin pitää tapahtua välittömästi, jotta molemmilla instansseilla olisi samat tiedot käytössä. Tietokannan kahdennus selvitettiin jo läpi tarkemmin alakohdassa 4.1.2. Rajapinnan tehtävänä onkin toimia käyttäjältä tulevien kyselyiden välittäjänä. Kyselyt ovat yleensä tietojen hakemista tietokannasta erilaisilla parametreilla tai tietojen lisäämistä ja päivittämistä tietokantaan. (Maksujärjestelmäprojektiryhmä, 2015)

Palveluiden tilattomuus mahdollistaa palveluiden helpon kahdentamisen tai monistamisen. Palvelimen tilattomuudella tarkoitetaan sitä, että käyttäjille ei avata istuntoa palvelimelle. Jokainen kysely, jonka käyttäjä tekee palveluun, sisältää kirjautumisessa saadun tokenin. Tämän tokenin avulla, tunnistetaan käyttäjä, sekä osataan rajata hänelle kuuluvat palvelut ja tiedot. Koska palvelimelle ei avata käyttäjäkohtaista istuntoa, voidaan käyttäjiltä tulevat kyselyt ohjata mille tahansa palvelinohjelmiston instanssille. Kyselyiden ohjaamisen tekee kuormantasaaja, joka tarkkailee palvelimien kuormitusta ja ohjaa kyselyt yleensä vähimmällä kuormalla olevalle palvelimelle. (Maksujärjestelmäprojektiryhmä, 2015) (Doglio, 2015)

5.3.2 Tilan tarkkailu

Palveluiden tilojen tarkkailulla pyritään huomaamaan mahdolliset poikkeukset tai palveluiden kaatumiset. Kuormantasaajan tehtävänä on muun muassa tarkastella, että palvelut ovat ylhäällä ja ohjata liikenne vain ylhäällä oleville palveluille. Esimerkiksi kahdennetun palvelun tapauksessa ohjataan kaikki liikenne suoraan toiselle palvelulle, jos toinen palveluista on alhaalla. Tämä on esitetty kuvassa 10. (Linux Virtual Server, 1998)

Tilaa tarkkaillaan myös ylläpidon toimesta, jonka tehtävänä on huolehtia järjestelmän toiminnasta. Järjestelmän toiminnasta saadaan parhaiten tietoja tarkkailemalla lokeja. Lokeja voidaan kirjoittaa eritasoisiksi, jolloin niiden tasosta voidaan päätellä lokin vakavuus. Esimerkiksi information-tason lokit ovat yleisiä järjestelmän tilaa kuvaavia lokeja, mutta warning- ja error-tasot tarkoittavat jo jonkinlaista häiriötä tai huomionarvoista asiaa. Järjestelmä on konfiguroitu niin, että error-tason lokeista lähtee automaattisesti tieto ylläpidolle. Ylläpito toimii tämän lokin perusteella ja korjaa tilanteen tai ilmoittaa siitä eteenpäin.



Kuva 10: Kuormantasaajan kyselyiden ohjaaminen, kun toinen kahdennetusta verkkopalvelusta on alhaalla.

5.3.3 Lokien hallinta

Lokien hallinta ja niiden tuottaminen on tärkeässä osassa palvelua. Lokien avulla saadaan kerättyä tärkeätä tietoa palvelun toiminnasta. Virhetilanteiden selvittämisessä lokit ovat entistäkin tärkeämmässä osassa. Lokien avulla saadaan selvitettyä, mitä virhetilanteen aikana on tapahtunut ja mikä sen on mahdollisesti aiheuttanut. Lokeilla tulee olla aina jokin syy, miksi niitä kerätään. (Valtiohallinnon tietoturvallisuuden johtoryhmä, 2009)

Lokeista ei kuitenkaan ole mitään hyötyä, jos kukaan ei niitä seuraa. Tästä syystä lokien seuraamiseen pitäisi olla prosessi, jossa ne käydään läpi, joko käsin tai automaattisesti. Etenkin vakavista virhetilanteista pitäisi tulla jonkinlainen hälytys ylläpitäjille, jotta virheisiin voidaan reagoida nopeasti. Palveluille on yleensä määritetty jonkinlainen SLA-taso, jonka puitteissa mahdolliset virheet pitäisi korjata. (Valtiohallinnon tietoturvallisuuden johtoryhmä, 2009)

Lokien keräämisessä pitää myös huomioida madolliset lait tai standardit, jotka voivat vaikuttaa lokien keräämiseen. Esimerkiksi henkilötietolakia tulee miettiä, jos lokit pitävät sisällään jotain henkilöiden yksilöllisiä tietoja. Maksujärjestelmien puolella on olemassa muun muassa PCI-DSS standardi, joka määrittelee, minkälaisia lokeja tulee kerätä ja miten näitä pitää hallita. Lisäksi on olemassa tietoturvallisuus standardeja, jotka määrittelevät lokien hallintaa. Tällaisia ovat esimerkiksi ISO 27001 (ISO, 2016) ja ISO 27002 (ISO, 2013). (Valtiohallinnon tietoturvallisuuden johtoryhmä, 2009)

Lokien säilyttämien ja arkistoiminen on mietittävä omiin tarpeisiin sopivaksi. Hyökkääjän yksi ensimmäisistä tehtävistä on poistaa lokeista omat jäljet. Tästä syystä

lokeja tulisi säilyttää keskitetyssä paikassa, jossa on rajattu pääsy. Lokien eheyttä voi parantaa luomalla lokeista tiivisteitä, jolloin muutokset lokeihin nähdään vertaamalla lokeista laskettua tiivistettä aiemmin laskettuun tiivisteeseen. Lokien arkistointiaika tulee myös jättää tarpeeksi pitkäksi. Mahdollisia hyökkäyksiä voi olla vaikeata havaita tai ne tullaan havaitsemaan myöhään. Tällöin lokien tulisi ulottua mahdollisimman pitkälle, jotta hyökkäyksen aiheuttamat vahingot saataisiin selville. (Valtiohallinnon tietoturvallisuuden johtoryhmä, 2009)

6. ARVIOINTI

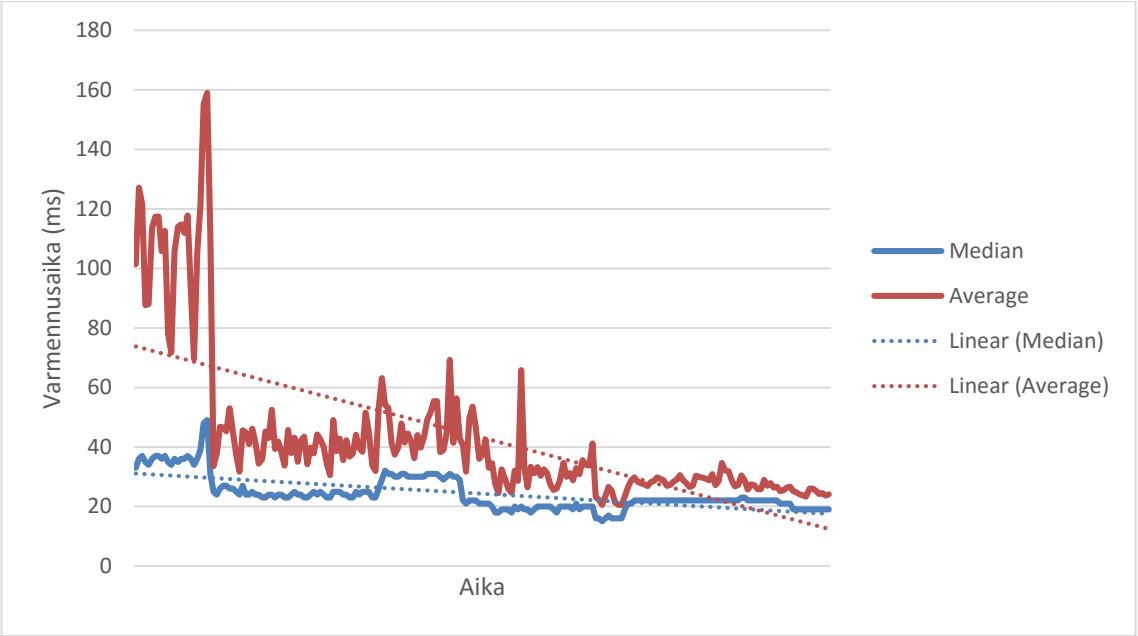
Tutkimuskohteena ollut maksujärjestelmä on tuotantokäytön aikana saanut useita päivityksiä. Päivityksillä on tuotu järjestelmään lisää vakautta, uusia ominaisuuksia sekä suorituskykyparannuksia. Järjestelmään on tehty myös tietoturva-auditointeja kolmannen osapuolen toimesta. Auditointien jälkeen on tehty parannusehdotuksien korjauksia. Tietoturvaa on parannettu muutoinkin entisestään ja sitä pidetään yllä mahdollisten haavoittuvuuksien estämiseksi. Järjestelmän ylläpidon huomioitaviin asioihin kuuluukin järjestelmän käyttämien ohjelmakirjastojen ja tekniikoiden seuraaminen, jotta saadaan tieto niihin kohdistuvista haavoittuvuuksista.

6.1 Suorituskyky

Suorituskykyparannuksilla on saatu käyttöliittymiä nopeammiksi ja sulavammiksi. Tämä parantaa järjestelmän käyttömukavuutta ja nopeuttaa erityisesti asiakaspalvelijoiden työtä. Suorituskykyä on saatu parannettua myös varmennuspalvelussa, jossa varmennuskyselyihin kuluvaa aikaa on saatu pienemmäksi, vaikka tietokannassa oleva tiedonmäärä on kasvanut huomattavasti. Varmennuskyselyihin kuluvaa aikaa seurataankin ylläpidon toimesta tarkemmin ja siitä kerätään päivittäin raporttia. Näiden raporttien perusteella on kuvassa 11 esitetty varmennuskyselyyn kuluvan ajan muuttuminen järjestelmän käyttöönottohetkestä aina tämän työn kirjoitushetkeen asti.

Kuten kuvasta huomataan, varmennusaikoja on saatu lyhennettyä huomattavasti tuotantoon viennin jälkeen. Punainen käyrä kuvaa yhden varmennuskyselyn keskimääräistä aikaa ja sininen käyrä on puolestaan varmennusajan mediaani. Tällä hetkellä varmennuskyselyyn saadaan muodostettua vastaus noin 30 millisekunnissa. Varmennusaikojen lyhentymiseen on vaikuttanut suurimmaksi osaksi juurikin tietokantakyselyiden optimointi. Varmennuskyselyihin tehdään reaaliaikainen käyttövaran laskenta, joka tarvitsee paljon tietoa tietokannasta. Toinen vaihtoehto olisi ollut laskea käyttövara valmiiksi tauluun, josta sen voisi tarkistaa. Tämä ei kuitenkaan huomioi käyttövaran muutoksia kovin nopeasti, jolloin käyttövara saattaisi ylittyä helposti.

Suurimmat suorituskykyparannukset ovat olleet tietokantakyselyiden optimointeja. Tietokanta on keskeisessä osassa järjestelmää ja sinne on haluttu viedä paljon laskentaa, jotta sitä ei tarvitse tehdä myöhemmässä vaiheessa. Tietokannat ovat myös yleensä vähällä laskentakuormalla ja niitä pidetään usein vain tietovarastona. Viemällä laskentaa tietokannalle, saadaan laskentakuormaa vähennettyä muilta.



Kuva 11: Varmennusaikojen muuttuminen järjestelmän toiminnan aikana.

Käyttöliittymien suorituskykyä on parannettu muun muassa hajauttamalla tiedon hakemista. Hajauttamisella tarkoitetaan, että kaikkia tietoja ei haeta ensimmäisessä sivun latauksessa, vaan osa tiedoista on piilotettu. Piilotettujen osioiden tiedot voidaan hakea vasta kun käyttäjä avaa osion. Tällöin harvemmin tarvittavia tietoja ei haeta aina, vaan tarvittaessa. Tämä nopeuttaa käyttöliittymän latausta sekä vähentää kuormaa taustapalveluissa, koska kyselyiden määrä näihin vähenee. Kyselyitä sivuttamalla on saatu isojen kyselyiden aikaa lyhennettyä, kun ei tarvitse odottaa koko kyselyn valmistumista. Esimerkiksi isojen listojen latausta voidaan tehdä osissa ja ladata seuraavan osion tiedot vasta kun käyttäjä on siirtymässä seuraavaan osioon.

Käyttöliittymille on tuotu myös paljon uusia ominaisuuksia. Erityisesti tämä näkyy loppukäyttäjille tarjottavassa käyttöliittymässä. Alussa tämä tarjosi loppukäyttäjille lähinnä tietojen tarkastelua sekä raporttien hakemista. Nykyään raportteja on laajennettu ja niiden hakemista nopeutettu. Tämän lisäksi loppukäyttäjät voivat tehdä vuorovaikutusta asiakaspalvelijoiden kanssa. Tämä tarkoittaa omien tilauksien tekemistä suoraan käyttöliittymältä ja tietojen muuttamista. Tilaukset menevät ensin asiakaspalveluun tarkastettavaksi, jonka jälkeen ne voidaan hylätä tai hyväksyä. Tämä vähentää asiakaspalvelijoiden työtä, kun tilausten tiedot on jo valmiiksi täytetty loppukäyttäjän toimesta.

6.2 Tietoturva

Tietoturvaa on pidetty aina mukana järjestelmän kehityksessä. Järjestelmään on tehty tietoturva-auditointeja ja niistä on päästy läpi. Joitakin parannusehdotuksia on tullut ja ne on huomioitu. Kriittisiä aukkoja ei kuitenkaan ole löytynyt ja kolmansien osapuolien

kirjastoista löytyviä aukkoja on pidetty silmällä ja tehty tarvittavia toimenpiteitä, jos ne ovat koskeneet tätä järjestelmää.

Kaikki tässä dokumentissa olevat kohdat eivät ole kuitenkaan yhden organisaation vastuulla. Esimerkiksi verkkoarkkitehtuurin suunnitteleminen ja ylläpito on suurimmaksi osaksi asiakkaan IT:n ja palveluntarjoajan vastuulla. Järjestelmän toimittajan puolelta on tullut itse arkkitehtuuriin vaikutusta, mutta verkkojen ylläpito ja laitteiden asetuksien huolehtiminen on palveluntarjoajan tehtävänä. Tarkempi toteutus on siis heidän päätettävissä neuvoteltujen SLA-tasojen toteuttamiseksi.

Tietokannan vastuu on osaksi järjestelmän toimittajan ja asiakkaan palveluntarjoajan vastuulla. Palveluntarjoajan tehtävänä on huolehtia kahdennuksista ja varmuuskopioista sekä tietokannan luonneista ja käyttäjäoikeuksista. Järjestelmän toimittajan vastuulla on puolestaan tietokannan päivittäminen, käyttäjähallinta ja oikeuksien myöntäminen sekä tietokannan tietojen eheyden ja salauksesta huolehtiminen.

Palvelinohjelmistojen toiminnan varmistaminen on puolestaan täysin organisaationi vastuulla. Kuormantasaajat kuuluvat kuitenkin palveluntarjoajalle ja he vastaavat niiden asetuksista. Palveluiden oikea toiminta ja mahdollisten hyökkäysten estäminen ja tiedon validointi kuuluvat omaan toteutukseen. Tähän kuuluvat myös palveluiden kahdennukset sekä tärkeimpänä hyvien ohjelmointikäytäntöjen noudattaminen. Hyvät ohjelmointikäytännöt vähentävät virheiden määrää ja tekevät ohjelmakoodista turvallisempaa. Tätä saadaan valvottua käyttämällä koodikatselmointeja tarkastamaan ohjelmakoodia ennen kuin se viedään versionhallintaan.

Koodikatselmoinnin hyöty on todettu erittäin hyväksi eikä siitä haluta luopua, vaikka se vie aina jonkin verran aikaa. Tämän käytännön avulla on kuitenkin löydetty paljon parannusehdotuksia sekä joitakin virheitä on saatu estettyä. Katselmoinnilla nähdään myös mitä ominaisuuksia muut ovat järjestelmään lisänneet.

6.3 Testaus

Ohjelmaan tehtävät muutokset ja lisäominaisuudet voivat aina rikkoa ohjelman toiminnallisuutta. Jotta muutoksia voisi viedä järjestelmään turvallisesti, pitää testausprosessien olla kunnossa. Testaukseen liittyy muun muassa yksikkö- ja järjestelmätestaus.

Yksikkötestit ovat ohjelman pienien osien testausta varten. Ne ajetaan automaattisesti, jos uutta ohjelmakoodia ollaan viemässä versionhallinnan päähaaraan. Näiden testien tarkoituksena on huomata virheitä, ennen kuin uusi versio ohjelmasta menee testaukseen. Näiden tekemisessä on huomioitu erityisesti kriittisten ominaisuuksien testaus. Esimerkiksi varmennuspalveluun liittyvään ohjelmakoodiin on tehty suurin osa

testeistä, jotta siihen tehtävät muutokset olisi helpompi tehdä ja sen toiminta saataisiin varmistettua muutoksien jälkeen.

Uudet ohjelmaversiot testataan ensin aina testausympäristössä. Testausympäristössä järjestelmää testataan kokonaisuudessaan, jolloin huomataan järjestelmien eri osien välillä tapahtuvat virheet. Järjestelmä testauksessa keskitytään erityisesti uusien ominaisuuksien ja korjattujen virheiden testaamiseen, mutta myös regressiotestausta tehdään järjestelmälle. Regressiotestauksella varmistetaan, että myös vanhat ominaisuudet toimivat eivätkä ne ole rikkoutuneet muutoksien takia. Regressiotestaukseen sopii parhaiten automaattiset työkalut, koska ne testaavat, että järjestelmä toimii niin kuin ennenkin.

6.4 Jatkokehitys

Järjestelmän jatkokehitykset turvaavat järjestelmän pitkäikäisyyden. Ilman jatkuvaa kehitystä, järjestelmä vanhenee ja muutoksien tekeminen menee vaikeammaksi. Esimerkiksi maksutavat ovat uudistumassa monin eri tavoin. Markkinoille tulee yhä enemmän mobiilimaksamiseen perustuvia maksutapoja. Mobiilimaksamisella voidaan tarkoittaa tavallisten maksukorttien lähimaksamisen siirtymistä mobiililaitteeseen tai QR-koodien skannaamista puhelimella.

Jatkokehityksien avulla mahdollistetaan järjestelmän modernisuus ja loppukäyttäjien tyytyväisyys. Loppukäyttäjien mielipiteet kuitenkin ratkaisevat, mitä maksutapaa he haluavat käyttää. Positiivisen kuvan palvelusta antaa monesti palvelun helppokäyttöisyys. Helppokäyttöisyys korostuu entisestään mobiililaitteissa, joissa on rajallisesti tilaa näytöllä. Uusien ominaisuuksien ja käyttöliittymien suunnittelussa pitääkin muistaa ottaa huomioon myös loppukäyttäjät, jotka oikeasti käyttävät palvelua.

7. YHTEENVETO

Maksujärjestelmäprojektilla oli hyvin tiukka aikataulu. Se oli myös jokseenkin haastava monien muuttuvien asioiden sekä järjestelmän kriittisyyden takia. Projektihenkilöiden määrä kasvoi verrattain suureksi, jolloin projektipäällikön organisointikyky tuli koetukselle. Tämä vaatii paljon projektipäälliköltä, jonka pitää osata jakaa ison järjestelmän tekemistä monelle tekijälle ja vielä olla tietoinen kaikkien tekemisistä. Ketterät projektikäytännöt auttoivat kuitenkin viemään projektin valmiiksi kaikilla tarvittavilla ominaisuuksilla aikataulun mukaisesti.

Kiireinen aikataulu on aina riski tietoturvan kannalta, koska silloin ei monesti ehditä miettimään kaikkia mahdollisia asioita, joita tulisi ottaa huomioon. Järjestelmän hyvä arkkitehtuurisuunnitelma mahdollisti tietoturvallisen järjestelmän tekemisen eikä siihen tarvinnut kiinnittää niin paljoa huomiota projektin edetessä. Tietenkin monet käytetyt ohjelmistokehykset auttavat tietoturvan huomioimisessa. Tästä huolimatta OWASP TOP 10 -tietoturvauhkaa pidettiin mielessä projektin aikana, jotta ne saadaan vältettyä lopullisessa järjestelmässä.

Kaikkiin tässä työssä mainittuihin tietoturva-alueisiin ei voitu itse vaikuttaa, koska asiakkaan konesalipalvelut oli hankittu kolmannen osapuolen kautta. Tästä syystä esimerkiksi verkkoarkkitehtuuri ja tietokannat olivat konesalitoimittajan huolehdittavana. Tietojen salaus tietokannassa ja palvelinohjelmistojen varmistaminen oli kuitenkin järjestelmän toimittajan tehtävänä. Se, mitä osa-alueita kuuluu järjestelmän toimittajan vastuulle, riippuu hyvin paljon asiakkaasta ja tuotantoympäristöstä. Joskus voidaan joutua vastaamaan kaikista näistä alueista. Nykyään ollaan myös menossa paljon pilvipalveluihin, jolloin palveluiden toiminnan varmistaminen on hyvin paljon pilvipalvelutoimittajan huolehdittavana. Tämä vähentää oman osuuden suunnittelua samoin kuin omista konesaleista siirryttäessä keskitettyihin konesaleihin.

Arvioinnissa käytiin läpi sitä, miten tietoturvallisuus saatiin hoidettua ja miten se on vaikuttanut järjestelmän suorituskykyyn. Monesti tiukat tietoturvallisuusprosessit heikentävät järjestelmän suorituskykyä. Tuotantoon viennin jälkeen suorituskykyä on kuitenkin saatu paljon parannettua. Suorituskykyparannukset on saatu toteutettua suurimmaksi osaksi tietokantakyselyitä optimoimalla. Tuotantodataa vasten on helpompaa tutkia suorituskyvyn ongelmakohtia, kun tiedetään miten paljon kyselyitä ajetaan ja minkälaisilla parametreilla.

Tietoturva saatiin toteutettua järjestelmässä sen vaatimalla tasolla. Tämän toteuttamisessa oli tietenkin omat riskinsä aikataulun vuoksi, mutta hyvä arkkitehtuuri

mahdollisti onnistumisen. Järjestelmään on tehty tietoturva-auditointeja ulkopuolisen toimijan suorittamana. Tähän mennessä on löytynyt ainoastaan suositeltavia asioita tietoturvan parantamisessa. Järjestelmä on myös ollut tuotantokäytössä jo melko pitkään eikä mitään tietoturvan kannalta kriittisiä asioita ole jouduttu paikkaamaan. Tietoturvahaavoittuvuuksia tarkkaillaan jatkuvasti, ja niihin reagoidaan asiaan kuuluvalla tavalla.

LÄHTEET

Cherry, D., 2012. *Securing SQL Server.* 2nd ed. Waltham: Syngress.

Cisco, 2008. *Campus Network for High Availability Design Guide.* [Online]
Available at:
http://www.cisco.com/c/en/us/td/docs/solutions/Enterprise/Campus/HA_campus_DG/hacampusdg.html
[Haettu 6 12 2015].

Cisco, 2014. *Cisco IOS Switching Services Configuration Guide, Release 12.2.* [Online]
Available at:
http://www.cisco.com/c/en/us/td/docs/ios/12_2/switch/configuration/guide/fswtch_c.html
[Haettu 21 11 2015].

Cisco, 2016. *Networking Basics: What You Need To Know.* [Online]
Available at:
http://www.cisco.com/cisco/web/solutions/small_business/resource_center/articles/connect_employees_and_offices/networking_basics/index.html
[Haettu 31 1 2016].

Convery, S., 2007. *Network Authentication, Authorization, and Accounting: Part One.* [Online]
Available at: http://www.cisco.com/web/about/ac123/ac147/archived_issues/ipj_10-1/101_aaa-part1.html
[Haettu 31 1 2016].

Doglio, F., 2015. *Pro REST API Development with Node.js.* 1st toim. La Paz, Canelones: Apress.

EMVCo, 2015. *EMVCo.* [Online]
Available at: http://www.emvco.com
[Haettu 7 11 2015].

F5, 2016. *Reverse Proxy.* [Online]
Available at: https://f5.com/glossary/reverse-proxy
[Haettu 16 1 2016].

Finanssialan keskusliitto, 2011. *EMV-maksupäätejärjestelmä toiminnallinen kuvaus (MPJ 042001).* [Online]
Available at:
https://www.fkl.fi/teemasivut/sepa/tekninen_dokumentaatio/Dokumentit/EMV_maksupaatejarjestelma.pdf
[Haettu 7 11 2015].

Git, 2016. *Git.* [Online]
Available at: https://git-scm.com/
[Haettu 2 4 2016].

Habas, D. & Sieber, J., 2006. *Background Patrol Read for Dell PowerEdge RAID Controllers.* [Online]
Available at: http://www.dell.com/downloads/global/power/ps1q06-20050212-Habas.pdf
[Haettu 13 4 2016].

Howes, T. A., 1995. *The Lightweight Directory Access Protocol: X.500 Lite,* Michigan: Center for Information Technology Integration.

Huston, T., 2016. *WHAT IS CODE REVIEW?.* [Online]
Available at: https://smartbear.com/learn/code-review/what-is-code-review/
[Haettu 2 4 2016].

Hu, V. C., Ferraiolo, D. F. & Kuhn, R. D., 2006. *Assessment of Access Control Systems,* Gaithersburg: National Institution of Standards and Technology.

ISO, 2013. *ISO/IEC 27002:2013.* [Online]
Available at: http://www.iso.org/iso/catalogue_detail?csnumber=54533
[Haettu 13 4 2016].

ISO, 2016. *ISO/IEC 27001 - Information security management.* [Online]
Available at: http://www.iso.org/iso/iso27001
[Haettu 13 4 2016].

Linux Virtual Server, 1998. *High Availability.* [Online]
Available at: http://www.linuxvirtualserver.org/HighAvailability.html
[Haettu 19 12 2015].

Maksujärjestelmäprojektiryhmä, 2015. *Maksujärjestelmän määrittely,* Tampere: Maksujärjestelmäprojektiryhmä.

Microsoft, 2001. *Virtual Private Networking.* [Online]
Available at: https://technet.microsoft.com/en-us/library/bb742566.aspx
[Haettu 19 12 2015].

Microsoft, 2014. *The OSI Model's Seven Layers Defined and Functions Explained.* [Online]
Available at: https://support.microsoft.com/en-us/kb/103884
[Haettu 31 1 2016].

Microsoft, 2016. *ENCRYPTBYASYMKEY (Transact-SQL).* [Online]
Available at: https://msdn.microsoft.com/en-us/library/ms186950(v=sql.120).aspx
[Haettu 16 1 2016].

Microsoft, 2016. *ENCRYPTBYCERT (Transact-SQL).* [Online]
Available at: https://msdn.microsoft.com/en-us/library/ms188061(v=sql.120).aspx
[Haettu 16 1 2016].

Microsoft, 2016. *ENCRYPTBYKEY (Transact-SQL).* [Online]
Available at: https://msdn.microsoft.com/en-us/library/ms174361(v=sql.120).aspx
[Haettu 16 1 2016].

Microsoft, 2016. *ENCRYPTBYPASSPHRASE (Transact-SQL).* [Online]
Available at: https://msdn.microsoft.com/en-us/library/ms190357(v=sql.120).aspx
[Haettu 16 1 2016].

Microsoft, 2016. *Overview of AlwaysOn Availability Groups (SQL Server).* [Online]
Available at: https://msdn.microsoft.com/en-us/library/ff877884.aspx
[Haettu 2 4 2016].

Microsoft, 2016. *So What Is Active Directory?.* [Online]
Available at: https://msdn.microsoft.com/en-us/library/windows/desktop/aa746492(v=vs.85).aspx
[Haettu 23 1 2016].

Microsoft, 2016. *Structured Query Language (SQL).* [Online]
Available at: https://msdn.microsoft.com/en-gb/library/windows/desktop/ms714670(v=vs.85).aspx
[Haettu 31 1 2016].

Microsoft, 2016. *Transact-SQL Reference (Database Engine).* [Online]
Available at: https://msdn.microsoft.com/en-us/library/bb510741.aspx
[Haettu 31 1 2016].

Morand, L. & Dekok, A., 2014. *Introduction to centralized Authentication, Authorization and Accounting (AAA) management for distributed IP networks.* [Online]
Available at: https://www.ietf.org/edu/tutorials/IETF89-Tutorial-AAA.pdf
[Haettu 13 4 2016].

National Institute of Standards and Technology, 2012. *NIST SP 800-30 Revision 1 Guide for Conducting Risk Assesments.* [Online]
Available at: http://csrc.nist.gov/publications/nistpubs/800-30-rev1/sp800_30_r1.pdf
[Haettu 7 11 2015].

OWASP, 2009. *Guide to Authorization.* [Online]
Available at: https://www.owasp.org/index.php/Guide_to_Authorization
[Haettu 6 2 2016].

OWASP, 2013. *OWASP Top Ten Project.* [Online]
Available at: https://www.owasp.org/index.php/Top10#OWASP_Top_10_for_2013
[Haettu 24 1 2016].

OWASP, 2015. *Access Control Cheat Sheet.* [Online]
Available at: https://www.owasp.org/index.php/Access_Control_Cheat_Sheet
[Haettu 6 2 2016].

OWASP, 2015. *REST Security Cheat Sheet.* [Online]
Available at: https://www.owasp.org/index.php/REST_Security_Cheat_Sheet
[Haettu 16 1 2016].

OWASP, 2016. *Input Validation Cheat Sheet.* [Online]
Available at: https://www.owasp.org/index.php/Input_Validation_Cheat_Sheet
[Haettu 16 1 2016].

OWASP, 2016. *OWASP.* [Online]
Available at: https://www.owasp.org/
[Haettu 24 1 2016].

OWASP, 2016. *Secure Coding Principles.* [Online]
Available at: https://www.owasp.org/index.php/Secure_Coding_Principles
[Haettu 31 1 2016].

Rich, B. R., 1995. *Clarence Leonard (Kelly) Johnson 1910-1990.* Washington D.C.: National Academies Press.

Rommann, R., 2014. *Cash Is Trash: The Future of Mobile Payment.* [Online]
Available at: http://www.forbes.com/sites/techonomy/2014/01/23/cash-is-trash-the-future-of-mobile-payment
[Haettu 5 3 2016].

Segan, S., 2015. *3G vs. 4G: What's the Difference?.* [Online]
Available at: http://www.pcmag.com/article2/0,2817,2399984,00.asp
[Haettu 31 1 2016].

Service Level Agreement Zone, 2015. *The Service Level Agreement.* [Online]
Available at: http://www.sla-zone.co.uk/
[Haettu 19 3 2016].

Singhal, A., Winograd, T. & Scarfone, K., 2007. *Guide to Secure Web Services.* [Online]
Available at: http://csrc.nist.gov/publications/nistpubs/800-95/SP800-95.pdf
[Haettu 21 2 2016].

Techopedia, 2016. *Keep It Simple Stupid Principle (KISS Principle).* [Online]
Available at: https://www.techopedia.com/definition/20262/keep-it-simple-stupid-principle-kiss-principle
[Haettu 31 1 2016].

Valtiohallinnon tietoturvallisuuden johtoryhmä, 2003. *Valtion tietohallinnon Internet-tietoturvallisuusohje.* [Online]
Available at: https://www.vahtiohje.fi/web/guest/1/2003-valtion-tietohallinnon-internet-tietoturvallisuusohje
[Haettu 23 11 2015].

Valtiohallinnon tietoturvallisuuden johtoryhmä, 2006. *Tunnistaminen julkishallinnon verkkopalveluissa.* [Online]
Available at: https://www.vahtiohje.fi/web/guest/12/2006-tunnistaminen-julkishallinnon-verkkopalveluissa
[Haettu 7 2 2016].

Valtiohallinnon tietoturvallisuuden johtoryhmä, 2009. *Lokiohje.* [Online]
Available at: https://www.vahtiohje.fi/web/guest/3/2009-lokiohje
[Haettu 7 2 2016].

Valtiohallinnon tietoturvallisuuden johtoryhmä, 2010. *Sisäverkko-ohje.* [Online]
Available at: https://www.vahtiohje.fi/web/guest/3/2010-sisaverkko-ohje
[Haettu 10 11 2015].

Vartiainen, T., 2011. *TLT-3301 Verkon tietoturva - Luento 10: Saatavuus ja kahdentaminen.* [Online]
Available at:
https://moodle2.tut.fi/pluginfile.php/113098/mod_resource/content/1/luennot-2011/TLT-3301-2011-luento10.pdf
[Haettu 6 12 2015].

Woodberg, B., 2011. *Defcon.* [Online]
Available at: https://www.defcon.org/images/defcon-19/dc-19-presentations/Woodberg/DEFCON-19-Woodberg-Network-Application-Firewalls.pdf
[Haettu 6 12 2015].